CS Exit Ethiopia

Questions & Worked Answers

Computer Science Exit Examination

184 complete questions

1 selected course | 18 November 2026

Selected courses

01. Computer Organization and Architecture

https://csexitethiopia.com

abelasnake2003@gmail.com | Telegram: @Abechu7

Course contents

Each course begins on a new page. PDF bookmarks provide question, study-note section and revision-topic navigation.

Computer Organization and Architecture 3

Reading the answers

Every question includes its answer, explanation, choice-by-choice contrasts and available evidence. Primary note proof quotes the study notes. Related reading and module passages are labelled separately; a module reference is not presented as primary note proof.

Computer Organization and Architecture

Questions & Worked Answers

184 questions

Question 1 | Q00029

Which one of the following is responsible for controlling the flow of bits between memory and controllers with no need for a constant intervention from the CPU?

A. Disk fragmenting

B. Memo, manager

C. Device controller

D. Direct memory access

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: Direct memory access

DMA transfers a block directly between device and memory; the CPU supplies setup information and receives completion notification, rather than moving every word (PDF pp.107-108). This precisely meets the absence of constant intervention. A generic device controller alone does not establish DMA capability.

Why each choice is right or wrong

A - Incorrect: Disk fragmenting is not the block-transfer mechanism described on pp.107-108.

B - Incorrect: The corrupted name “Memo, manager” does not identify the DMA mechanism on p.107.

C - Incorrect: An I/O controller can notify the CPU under interrupt I/O; that still leaves CPU data movement (p.106).

D - Correct: Direct transfers and only beginning/end CPU involvement are explicit on p.108.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

DMA lets a controller move a block directly between device and memory after CPU setup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
• Whether to read or write
• Device address
• Starting address of memory block for data
• Amount of data to be transferred The CPU carries on with other work.
• Thus DMA controller steals the CPU‟s work of I/O operation.
• The DMA module transfers the entire block of data
• One word at a time, directly to or from memory, without going through CPU.
When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Question sources

MOE Computer Science Exit Exam · 2016 · Question 29

Question 2 | Q00044

Which one of the following is a communication pathway connecting transmitting signals in binary digits between two or more devices?

A. Memory

B. Register

C. CPU

D. Bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: Bus

The common bus is a set of shared lines used to transfer binary information among registers (PDF p.40). The structural definitions distinguish storage and processing components from their communication mechanism (p.6).

Why each choice is right or wrong

A - Incorrect: Main memory stores data; it is an endpoint (p.6).

B - Incorrect: Registers store bits, rather than naming the pathway (p.35).

C - Incorrect: CPU processes and controls; it is a communicating component (p.6).

D - Correct: A bus supplies common bit lines and selection controls (p.40).

Primary note proof

Study-note section COA-S02

A bus is a set of common signal lines used to transfer binary information among connected units.

Study-note section COA01

The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

4.2 Bus and Memory Transfers
A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 44

Question 3 | Q00056

Which memory is characterized by the need for periodic change in order to maintain data storage?

A. Static RAM

B. Dynamic RAM

C. Disk

D. Tape

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Dynamic RAM

Interpret “periodic change” as periodic charge refreshing: capacitor charge leaks, so DRAM needs refresh (PDF p.80). This is different from SRAM power retention and nonvolatile secondary storage (p.91).

Why each choice is right or wrong

A - Incorrect: SRAM uses flip-flops and holds data while powered (p.80), without the stated charge refresh.

B - Correct: DRAM capacitor discharge explicitly requires periodic refreshing (p.80).

C - Incorrect: Disk is nonvolatile secondary storage (p.91), not capacitor-refresh RAM.

D - Incorrect: Tape is also nonvolatile secondary storage (p.91), not refresh RAM.

Primary note proof

Study-note section COA24

> SRAM uses powered bistable storage: fast and costly per bit, without periodic refresh.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

> DYNAMIC RAM (DRAM)
> is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.

STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 56

Question 4 | Q00060

In the ordinary HDD model, which statement is incorrect? Read/write refers to accessing sectors on the current track; seeking to another track is a separate operation.

A. Data is arranged in sectors within tracks

B. The head stays over the selected track while its sectors rotate beneath it

C. Its data is organized in tracks

D. It needs continuous power to keep bit contents

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: It needs continuous power to keep bit contents

An HDD records magnetic patterns in sectors arranged on tracks. During access to a selected track, rotation brings sectors under the head; seeking moves the head radially to another track. The medium retains its magnetic contents without continuous power, so the claimed power requirement is false. Power is needed to operate the drive, but that differs from power needed merely to retain the stored bits.

Why each choice is right or wrong

A - Incorrect: Sectors are subdivisions of tracks in the stated HDD organization.

B - Incorrect: This describes track-local access; a separate seek changes which track is under the head.

C - Incorrect: Magnetic disk data is organized on tracks, so this statement is true.

D - Correct: An HDD is nonvolatile: retaining stored patterns does not require continuous power.

Primary note proof

Study-note section COA28

An HDD stores magnetic patterns on tracks divided into sectors; reads/writes access addressed blocks while platters rotate.

Study-note section COA28

A head normally stays over a track while reading but moves radially for seeks.

Study-note section COA28

It does not need continuous power to preserve stored bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 60

Question 5 | Q00093

Which listed circuit category requires stored state so that output can depend on past input behavior as well as present inputs?

A. Combinational circuit

B. Sequential circuits

C. A two-input AND gate with no feedback

D. A combinational multiplexer with no storage

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer B: Sequential circuits

A sequential circuit contains state that preserves information about prior behavior. A combinational circuit computes its outputs from current inputs after propagation delay. The specified AND gate and multiplexer are combinational examples without storage or feedback. They therefore do not require remembered state, while the sequential-circuit category does. This distinction concerns logical state, not whether physical propagation takes time.

Why each choice is right or wrong

A - Incorrect: Combinational output depends on current inputs rather than stored history.

B - Correct: Sequential circuits retain state and can make output depend on past behavior.

C - Incorrect: The specified AND gate computes a truth function of its present inputs without storage.

D - Incorrect: The specified multiplexer selects a present data input; it has no stored logical state.

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> Adders, multiplexers, demultiplexers, encoders and decoders have no stored logical state.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Computer Organization and Architecture module | PDF page 10 | lines 22-27

1.4 Combinational Circuits
• A combinational circuit is an interconnected set of gates whose output at any time is a function only of the input at that time. As with a single gate, the appearance of the input is followed almost immediately by the appearance of the output, with only gate delays. In general terms, a combinational circuit consists of n binary inputs and m binary outputs.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 93

Question 6 | Q00134

In computers, subtraction is carried out generally by _____.

A. 1's complement method

B. 2's complement method

C. Signed magnitude method

D. BCD subtraction method

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer B: 2's complement method

Subtraction is described as complementation plus addition: add one to the inverted subtrahend, then add that two’s complement to the minuend (PDF p.42). Hence R1 + (~R2+1) represents R1−R2 within the register width.

Why each choice is right or wrong

A - Incorrect: One’s complement alone omits the added 1 in p.42’s subtraction procedure.

B - Correct: The inversion-plus-one subtrahend is exactly two’s complement (p.42).

C - Incorrect: The cited procedure does not use separate sign and magnitude arithmetic.

D - Incorrect: The cited general register subtraction is not BCD digit subtraction.

Primary note proof

Study-note section COA09

> Compute A−B as A+(two’s complement of B), retaining n result bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 42 | lines 21-25

> Substruction is most often implemented through complementation and addition.
> Example, ¯
> ¯is the symbol for the 1’s complement of R2. Adding 1 to the 1’s complement produces the 2’s complement. Adding the contents of R1 to the 2’s complement of R2 is equivalent to R1-R2.

Question sources

Addis Ababa University mock exam · Question 36

Computer Science Exit Exam · Sene 2016 · Question 20-3

Computer Science Exit Exam practice collection · Question 239

Question 7 | Q00135

Pick the different _

A. Decoder

B. Encoder

C. Multiplexer

D. Shift Register

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer D: Shift register: the sequential circuit.

COA05 distinguishes combinational circuits from sequential storage. Decoder, encoder and multiplexer are explicitly combinational; the shift register retains state and therefore belongs to the sequential class. The image confirms the four option texts. Using the state criterion provides the technically supported odd-one-out distinction without relying on the source emphasis.

Why each choice is right or wrong

A - Incorrect: A decoder is combinational and has no stored logical state in this circuit classification.

B - Incorrect: An encoder is also combinational, grouping it with the decoder and multiplexer.

C - Incorrect: A multiplexer selects among present inputs and is explicitly a combinational circuit.

D - Correct: A shift register retains and shifts stored state, distinguishing it from the three combinational devices.

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 33 | lines 7-14

> 3.7 Decoders
> • A decoder is a combinational circuit with a number of output lines, only one of which is selected at any time, depending on the pattern of input lines.
> • In general, a decoder has n inputs and 2n outputs.

• Decoders find many uses in digital computers.
• One example is address decoding.
• The other is binary-to- octal conversion.

Computer Organization and Architecture module | PDF page 34 | lines 3-8

3.8 Encoders
• An encoder is a digital circuit that performs the inverse operation of a decoder.
• An encoder has 2n
(or less) input lines and n output lines. The output lines generate
the binary code corresponding to the input value.
• An example of an encoder is the octal-to-binary encoder.

Computer Organization and Architecture module | PDF page 31 | lines 12-16

3.5 Multiplexers
• Multiplexer is a combinational circuit that receives binary information from one of 2n input data & directs to one output.
• The multiplexer connects multiple inputs to a single output.
• At any time, one of the inputs is selected to be passed to the output.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Computer Organization and Architecture module | PDF page 36 | lines 2-9

Shift Register
• A shift register accepts and/or transfers information serially.
• Data are input only to the leftmost flip-flop.
• Shift registers can be used to interface to serial I/O devices.
• With each clock pulse, data are shifted to the right, one position, and the rightmost bit

is transferred out.
• In addition, they can be used within the ALU to perform logical shift and rotate functions.

Question sources

Addis Ababa University mock exam · Question 37

Question 8 | Q00136

An operation performed on the contents of a register is known as

A. Instruction code

B. Micro-operation

C. Accumulator

D. Register

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Micro-operation

An elementary operation on information stored in one or more registers is explicitly defined as a micro operation (PDF p.38). An instruction code tells the computer what operation to execute, while registers are the storage objects (pp.35,47).

Why each choice is right or wrong

A - Incorrect: An instruction code specifies work; it is not the elementary register operation itself (p.47).

B - Correct: Matches the exact definition and examples shift, count, clear, load (p.38).

C - Incorrect: Accumulator denotes a processing register, not the operation.

D - Incorrect: A register stores the information being operated on (p.35).

Primary note proof

Study-note section COA11

A microoperation is one elementary operation on register contents.

Study-note section COA16

An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 38 | lines 8-12

Digital modules are best defined by the registers they contain and the operations that are performed on the data stored in them. The operations executed on data stored in registers are called micro operations. A micro operation is an elementary operation performed on the information stored in one or more registers.
Examples of micro operation are shift, count, clear, and load.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Question sources

Addis Ababa University mock exam · Question 38

Question 9 | Q00137

Which one is true about RISC computers?

A. RISC are with few numbers of registers when compared to CISC

B. RISC use complex and efficient machine instructions when compared to CISC

C. RISC is with hardwired control unit with pipelining processing

D. RISC uses extensive addressing capabilities for memory operations

E. None

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: RISC is with hardwired control unit with pipelining processing

RISC characteristics explicitly include hardwired control, many registers, few addressing modes and an efficient pipeline (PDF p.77). These jointly support C and contradict the reversed CISC-like descriptions.

Why each choice is right or wrong

A - Incorrect: A reverses the relatively large register count on p.77.

B - Incorrect: Complex instructions are the CISC characteristic (p.78), whereas RISC has few, easily decoded instructions (p.77).

C - Correct: Both hardwired control and efficient pipeline are explicitly listed (p.77).

D - Incorrect: RISC has relatively few addressing modes and memory access limited to loads/stores (p.77).

E - Incorrect: C is supported, so “None” fails.

Primary note proof

Study-note section COA-S04

> The textbook RISC design pairs a relatively large register set, load/store access and easily decoded instructions with hardwired rather than microprogrammed control and an efficient pipeline.

Study-note section COA20

> CISC commonly supplies more complex operations, richer addressing modes and variable lengths.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Question sources

Addis Ababa University mock exam · Question 39

Question 10 | Q00138

Which of the following is not a basic element within the microprocessor?

A. Microcontroller

B. Arithmetic logic unit (ALU)

C. Register array

D. Control unit

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Microcontroller

The module divides the CPU into the register set, ALU and control unit (PDF p.57). Treating microprocessor as the CPU implementation, A is the only choice outside that list; the module does not separately define microcontrollers.

Why each choice is right or wrong

A - Correct: Microcontroller is not one of the three CPU components listed on p.57.

B - Incorrect: ALU performs the micro-operations (p.57).

C - Incorrect: The register set stores intermediate data (p.57).

D - Incorrect: Control unit supervises transfers and ALU operations (p.57).

Primary note proof

Study-note section COA01

> The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Addis Ababa University mock exam · Question 40

Question 11 | Q00139

After CPU setup, which transfer method moves a block between an I/O device and memory without the CPU copying each data word?

A. Software interrupts

B. Interrupt-driven I/O

C. Polled I/O

D. Direct memory access (DMA)

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Direct memory access (DMA)

DMA assigns the block movement to a controller after the CPU configures the transfer. Polling and ordinary interrupt-driven I/O still make the CPU execute item transfers. A software interrupt is an instruction-requested control transfer to a handler, not a block-copy engine. CPU setup and completion handling remain part of DMA; bypass refers specifically to avoiding CPU copying for every word.

Why each choice is right or wrong

A - Incorrect: A software interrupt requests handler execution; it does not itself move a block directly between device and memory.

B - Incorrect: The interrupt handler still uses CPU execution to service the device and move items.

C - Incorrect: The CPU checks status and moves items itself in polled I/O.

D - Correct: The DMA controller performs the block transfer after setup, avoiding per-word CPU copies.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA19

> A software interrupt is deliberately requested by an instruction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that
> allow data to be sent directly from an attached device (such as a disk drive) to the memory on

the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
• Whether to read or write
• Device address
• Starting address of memory block for data
• Amount of data to be transferred The CPU carries on with other work.
• Thus DMA controller steals the CPU‟s work of I/O operation.
• The DMA module transfers the entire block of data
• One word at a time, directly to or from memory, without going through CPU.
When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically

- I/O module does not inform CPU directly
- I/O module does not interrupt CPU

Question sources

Addis Ababa University mock exam · Question 41

Question 12 | Q00140

In the simple CPU-master bus model, which bus carries instruction or operand values in both directions during memory reads and writes?

A. Address bus

B. Control bus

C. Data bus

D. None of the above

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Data bus

Data signals carry the values transferred between processor and memory. A read brings those values toward the CPU, while a write sends them toward memory, so the data bus serves both directions. Address signals select a location. Control signals coordinate operations and can include signals in both directions, but the question specifically asks about instruction or operand values rather than aggregate control direction.

Why each choice is right or wrong

A - Incorrect: The address bus selects locations rather than carrying payload values.

B - Incorrect: Control signals coordinate transfers rather than carrying the instruction or operand payload.

C - Correct: The data bus carries values toward the CPU on reads and toward memory on writes.

D - Incorrect: The data bus supplies the requested role.

Primary note proof

Study-note section COA12

> Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Study-note section COA12

> In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

> • A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
> • A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 102 | lines 20-24

> I/O module must have the capability to engage in communication with the CPU and external device. Thus CPU communication involves
> Command decoding: The I/O module accepts commands from the CPU carried on the control bus.
> Data: data are exchanged between the CPU and the I/O module over data bus

Computer Organization and Architecture module | PDF page 103 | lines 2-9

> Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
> Address recognition:
> just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Question sources

Addis Ababa University mock exam · Question 42

Question 13 | Q00365

Which specific instruction-processing technique overlaps fetch or decode of a later instruction with execution of an earlier instruction?

A. Superscalar processing

B. None of the above

C. Parallel Computation

D. Pipe-lining

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Pipe-lining

Fetching or decoding one instruction while another is being executed overlaps different instruction stages, which is pipelining. Superscalar processing instead permits issuing multiple instructions to execution resources during a cycle. These techniques can coexist, but the specific fetch/decode-versus-execute overlap identifies the pipeline mechanism rather than the processor's issue width.

Why each choice is right or wrong

A - Incorrect: Superscalar processing names multiple-issue execution resources; the stated overlap of different stages specifically describes pipelining.

B - Incorrect: Pipelining is an available matching choice.

C - Incorrect: Parallel computation is broader simultaneous processing (p.114); it does not identify the staged instruction mechanism as precisely as D.

D - Correct: The instruction pipeline reads consecutive instructions while previous ones execute (p.117).

Primary note proof

Study-note section COA35

> "Fetch/decode one instruction while executing another" describes pipelining, whereas issuing several instructions in one cycle requires multiple-issue resources.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science model exam · Question 96

University of Gondar model exam · Question 96

Question 14 | Q00366

Which instruction class, where supported, bypasses the immediately following instruction when its condition is met?

A. Branch instruction

B. Return instruction

C. Skip instruction

D. Jump instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Skip instruction

A SKIP instruction conditionally bypasses the next instruction. That differs from an ordinary branch, which chooses a target, and from RETURN, which restores a saved return point. The original phrase jump to the next instruction did not express bypassing it; the revised wording states that intended control-flow effect precisely. The classification is based on behavior rather than an unstated machine-specific opcode.

Why each choice is right or wrong

A - Incorrect: A branch chooses a target rather than naming the specific next-instruction-bypass class.

B - Incorrect: RETURN restores a saved return point.

C - Correct: SKIP is the conditional next-instruction-bypass operation described.

D - Incorrect: A general jump chooses a target rather than naming this specific conditional operation.

Primary note proof

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Study-note section COA19

> CALL saves a return point; RETURN restores it.

Study-note section COA19

> SKIP, where supported, bypasses the next instruction when its condition holds.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

> 1. Fetch an instruction from memory
> 2. Decode the instruction
> 3. Read the effective address from memory if the instruction has an indirect address
> 4. Execute the instruction
> Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3.

Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 52 | lines 2-6

Table 5-2: Basic computer instructions
Arithmetic, logical and shift instructions provide computational capabilities for processing the type of data that the user may wish to employ. The bulk of the binary information in a digital computer is stored in memory, but all computations are done in processor registers. Therefore, the user must have the capability of moving information between these two units

Question sources

Computer Science model exam · Question 97

University of Gondar model exam · Question 97

Question 15 | Q00367

An interrupt that is caused by register overflow, attempt to divide by zero, an invalid operation code, stack overflow can be?

A. None of the above

B. Software interrupt

C. External interrupt

D. Internal interrupt

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Internal interrupt (synchronous exception).

COA19 classifies divide errors and illegal opcodes as synchronous internal exceptions because they arise from the executing instruction. Register or stack overflow belongs to that category when the machine traps it. Overflow does not invariably interrupt: ordinary flags may merely record it. Deliberate software requests and asynchronous external device events describe different causes.

Why each choice is right or wrong

A - Incorrect: The internal-exception category directly covers these instruction-related faults when trapping is enabled.

B - Incorrect: A software interrupt is deliberately requested by an instruction, rather than caused by an execution fault.

C - Incorrect: External interrupts originate from devices or timers, rather than the faulting instruction.

D - Correct: These execution-generated faults are internal exceptions; overflow trapping still depends on the architecture.

Primary note proof

Study-note section COA19

> External interrupts arise from devices or timers; synchronous internal exceptions arise from the executing instruction, such as divide errors, illegal opcodes or access violations.

Study-note section COA19

> A software interrupt is deliberately requested by an instruction.

Study-note section COA19

> Arithmetic overflow trapping depends on architecture and instruction; ordinary overflow flags need not trigger an interrupt.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

> with the higher priority first.
> Interrupt Driven I/O.
> Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for

coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Question sources

Computer Science model exam · Question 98

University of Gondar model exam · Question 98

Question 16 | Q00368

Computer address bus is

A. Unidirectional

B. None of the above

C. Multidirectional

D. Bidirectional

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Unidirectional

For the CPU-initiated memory operations described, CPU-held address registers supply the memory address (PDF pp.82-83 and 70). Thus the address path runs CPU→memory; this is a direction within that model, not a rule excluding DMA bus masters.

Why each choice is right or wrong

A - Correct: CPU loads MAR to select memory; the addressed memory does not return an address (pp.82-83).

B - Incorrect: The described CPU-address path supplies a matching option A.

C - Incorrect: The module's CPU-memory operation does not require simultaneous multidirectional address flow.

D - Incorrect: Read/write data can go both ways, but the initiating CPU supplies the address.

Primary note proof

Study-note section COA12

> In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 83 | lines 11-14

> The time that elapses between the initiation of an operation and completion of that operation is called memory access time. The minimum time delay between two successive memory operations is called memory cycle time. The cycle time is usually slightly lower than the access time.

Computer Organization and Architecture module | PDF page 83 | lines 2-7

> During a “memory cycle“ n bits of data are transferred between main memory and CPU. This transfer takes place over the processor bus, which has k address lines and n data lines. The CPU initiates a memory operation by loading the appropriate data into registers MDR and MAR, and setting either Read or Write memory control line to 1. When the required operation is completed the memory control circuitry sends Memory Function Completed (MFC) signal to CPU.

Question sources

Computer Science model exam · Question 99

University of Gondar model exam · Question 99

Question 17 | Q00369

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part specifies the operation to be performed and the second specifies .

A. address

B. output

C. Input

D. operand

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: address

The module reproduces this two-part instruction description: operation first, address second; that address locates the operand (PDF p.47). The operand itself and its address must not be conflated.

Why each choice is right or wrong

A - Correct: The second field specifies an address (p.47).

B - Incorrect: Output is not the second field in this format.

C - Incorrect: Input is not the second field in this format.

D - Incorrect: The operand is fetched using the address; it is not generally embedded in this field (p.47).

Primary note proof

Study-note section COA-S03

> In the textbook single-accumulator organization, a memory-reference instruction has an operation part and an address part.

Study-note section COA-S03

> The address locates a memory operand; the operation combines that operand with the accumulator as required.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

> The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one

section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Question sources

Computer Science model exam · Question 100

University of Gondar model exam · Question 100

Question 18 | Q00370

Which of the following computer memory is fastest?

A. RAM

B. Register

C. None of the above

D. Hard disk

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Register

The hierarchy figure on PDF p.79 places registers before cache, RAM and disk. Register organization also avoids time-consuming memory accesses (p.58). Although p.84 calls cache the fastest memory-hierarchy component, it qualifies cache as approaching CPU component speed; registers are the CPU storage meant here.

Why each choice is right or wrong

A - Incorrect: RAM requires memory access, avoided by retaining intermediate values in registers (p.58).

B - Correct: Registers are the fastest offered CPU storage tier in Figure 6.1 (p.79).

C - Incorrect: A listed choice fits.

D - Incorrect: Disk lies at the slow, large secondary-storage end of the hierarchy (p.79).

Primary note proof

Study-note section COA23

A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

6.1 Memory Hierarchy:
A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 58 | lines 4-11

In the programming examples of the previous chapter, we have shown that memory locations are needed for storing pointers, counters, return addresses, temporary results and partial product during multiplication. Having to refer to memory locations for such applications is time consuming because memory access is the most time consuming operation in a computer. It is more convenient and more efficient to store these intermediate values in processor register. When a large number of registers are included in the CPU, It is most efficient to connect them through a common bus system. The registers communicate with each other not only for direct data transfers, but also while performing various micro operations.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science model exam · Question 101

University of Gondar model exam · Question 101

Question 19 | Q00539

It is a method used to synchronize the I/O device with the microprocessor

A. Interrupt

B. Handshaking

C. Interface

D. None

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Handshaking.

COA31 describes asynchronous handshaking as an exchange of request/ready and acknowledgement signals. The sender presents data, the receiver accepts it, and the feedback coordinates transfer despite unequal device and processor speeds. An interrupt requests service but does not itself name this two-sided transfer protocol. An interface is the broader connection exposing the signals and registers.

Why each choice is right or wrong

A - Incorrect: Interrupts request processor attention; they do not specifically identify the request-and-acknowledgement transfer protocol.

B - Correct: Request and acknowledgement signals synchronize the sender and receiver despite unequal operating speeds.

C - Incorrect: An interface provides the connection; handshaking names the synchronization method implemented through it.

D - Incorrect: Handshaking is an explicitly supported synchronization method, so an affirmative option fits.

Primary note proof

Study-note section COA31

> Asynchronous handshaking uses request/ready and acknowledgement signals to tolerate unequal speeds: sender presents data and requests, receiver accepts and acknowledges, then each withdraws its signal.

Study-note section COA31

> A single strobe lacks this feedback.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 101 | lines 2-9

> Asynchronous data transfer
> The internal operation in digital system is synchronized by means of clock pulse supplied by a common pulse generator. Two units such as cpu and an I/O are designed independently of each other, if the register in the interface share a common clock with the cpu registers, the transfer between the two units is called to be synchronous.in most case the internal timing in each unit is independent from the other in that each uses its own private clock for internal register.in such cases the two units are said to be asynchronous to each other. This approach is widely used in most computers systems.

Computer Organization and Architecture module | PDF page 101 | lines 10-16

> 7.4 Input/output Problems
> Wide variety of peripherals and delivering different amounts of data per second Work at different speeds Send/receive data in different formats all slower than CPU and RAM. Hence I/O modules are used as a solution.
> Input/output Module: It is the entity within a computer that is responsible for the control of one or more external devices and for the exchange of data between those devices and main memory and/or CPU.

Question sources

Computer Science model exam · 2015 · Question 8

Question 20 | Q00542

Which one of the following instructions is incorrect?

A. MOV AL, BP

B. INC CX

C. DEC CX

D. All

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: MOV AL, BP is invalid.

Under the ordinary Intel forms specified in COA22, AL is an 8-bit destination while BP is a 16-bit source. MOV does not implicitly resize these operands, so MOV AL, BP is invalid. INC CX and DEC CX are legal 16-bit operations. Consequently the all-invalid alternative fails because two listed instructions are valid.

Why each choice is right or wrong

A - Correct: The 8-bit AL and 16-bit BP widths do not match for this MOV instruction.

B - Incorrect: INC CX legally increments a 16-bit register under the stated legacy x86 assumptions.

C - Incorrect: DEC CX legally decrements a 16-bit register under the stated legacy x86 assumptions.

D - Incorrect: INC CX and DEC CX are valid, so the instructions are not all incorrect.

Primary note proof

Study-note section COA22

> MOV AL,BP is invalid because operand widths differ (8 versus 16 bits).

Study-note section COA22

> INC CX and DEC CX are valid word operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

> Data transfer instruction.
> Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

> Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
> When the address field specifies a processor register, the instruction is said to be in the register mode.
> Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
> Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Computer Organization and Architecture module | PDF page 38 | lines 8-12

Digital modules are best defined by the registers they contain and the operations that are performed on the data stored in them. The operations executed on data stored in registers are called micro operations. A micro operation is an elementary operation performed on the information stored in one or more registers.
Examples of micro operation are shift, count, clear, and load.

Computer Organization and Architecture module | PDF page 43 | lines 3-12

R3←R1 + R2 Contents of R1 plus R2 transferred to R3
R3←R1 -R2 Contents of R1 minus R2 transferred to R3
R2←$\bar{}$Complement the contents of R2 (1's complement)
R2←$\bar{}$+ 1 2's complement the contents of R2 (negative)
R3←R1 +$\bar{}$+ 1 R1 plus the 2's complement of R2 (substruction)
R1←R1 + 1 Increment the contents of R1 by one
R1←R1- 1 Decrement the contents of R1 by one
Table 4.2: Arithmetic Micro operation
The increment and decrement micro operations are symbolized by plus one and minus- one operation, respectively.

Question sources

Computer Science model exam · 2015 · Question 11

Question 21 | Q00544

A register used to indicate the condition of the microprocessor is

A. SI

B. FLAG

C. DI

D. EIP

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: FLAG register.

COA21 states that FLAGS/EFLAGS/RFLAGS report processor status. This is the register family indicated by the option FLAG. SI and DI are general-purpose registers with conventional index roles, and EIP identifies the instruction position. Neither the index registers nor the instruction pointer is the processor condition/status register requested.

Why each choice is right or wrong

A - Incorrect: SI has an index/general-purpose role rather than collecting the processor condition flags.

B - Correct: FLAG denotes the status-register family reporting processor conditions such as arithmetic status.

C - Incorrect: DI has an index/general-purpose role rather than serving as the processor status register.

D - Incorrect: EIP identifies instruction position; it does not contain the collection of processor condition flags.

Primary note proof

Study-note section COA21

> FLAGS/EFLAGS/RFLAGS report status; IP/EIP/RIP identifies instruction position.

Study-note section COA21

> In Intel x86 notation, AX/BX/CX/DX, BP/SP and SI/DI are 16-bit general-purpose registers, with conventional special roles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

> Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction

There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science model exam · 2015 · Question 13

Question 22 | Q00545

Which one of the following instructions is correct

A. MOV DS, 20

B. MOV ES, 40

C. MOV SS, 60

D. None

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: None of these immediate-to-segment MOV forms is valid.

COA22 excludes a direct immediate-to-segment-register MOV in the assumed legacy x86 syntax. DS, ES and SS are segment registers, and each listed source is an immediate number. A permitted sequence first loads the value into a suitable general register and then moves that register to the segment register. Therefore all three displayed forms fail.

Why each choice is right or wrong

A - Incorrect: MOV DS, 20 attempts the unsupported direct transfer of an immediate into a segment register.

B - Incorrect: MOV ES, 40 also attempts an unsupported immediate-to-segment-register transfer in the stated syntax.

C - Incorrect: MOV SS, 60 has the same unsupported direct immediate source for a segment destination.

D - Correct: All three proposed segment-register instructions require an intermediate register or supported memory source.

Primary note proof

Study-note section COA22

Segment-register MOV has register/memory forms, not a direct immediate-to-segment form: load an immediate into AX, then MOV DS,AX.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

> Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
> When the address field specifies a processor register, the instruction is said to be in the register mode.
> Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
> Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

> Data transfer instruction.
> Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Question sources

Computer Science model exam · 2015 · Question 14

Question 23 | Q00546

A register used to address the next instruction in a section of memory is

A. RSP

B. CS

C. RIP

D. SS

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: RIP, the instruction pointer.

COA21 identifies IP/EIP/RIP as the instruction-position register family; COA14 identifies the PC role as locating the next instruction to fetch. RIP is the corresponding 64-bit x86 instruction pointer. CS selects the code segment and SS the stack segment, while RSP is the stack pointer. A segment selector alone does not identify the next instruction offset.

Why each choice is right or wrong

A - Incorrect: RSP tracks the stack, not the next instruction position within the instruction stream.

B - Incorrect: CS identifies the code segment; the instruction pointer supplies the instruction position within it.

C - Correct: RIP is the x86 instruction pointer used to identify the next instruction position.

D - Incorrect: SS identifies the stack segment rather than the location of the next instruction.

Primary note proof

Study-note section COA-S05

> In Intel 64-bit mode, RIP holds the offset of the next instruction to execute.

Study-note section COA-S05

> RSP is the 64-bit stack pointer, extending the ESP/SP stack role.

Study-note section COA-S05

> These pointers are distinct from CS and SS, which are code and stack segment registers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 69 | lines 15-23

> 5.15.4 Zero-Address Instruction
> A stack-organized computer does not use an address field for instructions ADD and MUL.
> The PUSH and POP instructions, however, need an address field to specify the operand that

communicates with the stack. The following program shows how X = (A+B) * (C+D) will be written for a stack-organized computer (TOS stands for top-of-stack)
PUSH A TOS← A, PUSH B TOS ←B, ADD TOS ←(A + B),
PUSH C TOS ←C
PUSH D TOS ←D, ADD TOS ←(C + D) , MUL TOS ←(C + D) (A + B),
POP X M[X] ←TOS.

Question sources

Computer Science model exam · 2015 · Question 15

Question 24 | Q00562

The Intel's first true 32-bit microprocessor is

A. Pentium

B. 80486

C. 80286

D. 80386

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer D: 80386, within the offered x86 lineage.

COA38 explicitly dates 32-bit x86 execution to the 80386 in 1985. It precedes the 80486 and Pentium, while the 80286 is a 16-bit predecessor. The offered choices therefore select 80386. This conclusion concerns the x86 sequence represented by the options; it should not be expanded into a claim about every historical Intel processor family.

Why each choice is right or wrong

A - Incorrect: Pentium follows the 80386; it was not the first 32-bit member among these x86 choices.

B - Incorrect: The 80486 follows the 80386, which had already introduced 32-bit x86 execution.

C - Incorrect: The note identifies the 80286 as a 16-bit predecessor, rather than the 32-bit introduction.

D - Correct: The 80386 introduced 32-bit x86 execution before the other offered 32-bit processors.

Primary note proof

Study-note section COA38

Intel’s 80386 introduced 32-bit x86 execution in 1985, before 80486/Pentium; the 80286 was a 16-bit predecessor.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science model exam · 2015 · Question 31

Question 25 | Q00569

The content of BH is 10, AL is 8, CL is 4 how much is the content of AX after MUL BH

A. 40

B. 80

C. 32

D. 320

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: AX = 80.

The operand BH is a byte register. COA22 states that unsigned byte MUL multiplies AL by the explicit byte operand and stores the product in AX. Substituting the given values yields 8 × 10 = 80. CL is not an operand of MUL BH; multiplying by its value would change the instruction being evaluated.

Why each choice is right or wrong

A - Incorrect: 40 multiplies BH by CL, but byte MUL uses AL as its implicit multiplicand.

B - Correct: AL × BH equals 8 × 10 = 80, written into AX.

C - Incorrect: 32 multiplies AL by CL, whereas the explicit operand is BH, not CL.

D - Incorrect: 320 incorrectly includes CL as a third factor; MUL BH multiplies only AL and BH.

Primary note proof

Study-note section COA22

> Unsigned MUL with a byte operand multiplies AL by that byte and writes AX.

Study-note section COA22

> Thus AL=8 and BH=10 gives AX=80 after MUL BH; CL is irrelevant.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

> Data Manipulation
> Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
> 1. Arithmetic instruction
> 2. Logical and bit manipulation
> 3. Shift instruction
> a. Arithmetic Instruction
> The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

> Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Question sources

Computer Science model exam · 2015 · Question 38

Question 26 | Q00573

Which legacy x86 instruction is arithmetic rather than a data-transfer or operand-exchange operation?

A. MOVS

B. LODS

C. XCHG

D. INC

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: INC

MOVS and LODS are string-transfer operations, and XCHG exchanges operand values. INC instead increments an operand, making it arithmetic. The revised stem states the classification criterion, so different no longer depends on whether the reader groups string operations, operand counts or another property. This retains the original instruction set and intended comparison while giving every choice a definite classification.

Why each choice is right or wrong

A - Incorrect: MOVS transfers string data.

B - Incorrect: LODS loads string data.

C - Incorrect: XCHG exchanges operands rather than performing arithmetic.

D - Correct: INC increments an operand arithmetically.

Primary note proof

Study-note section COA22

> MOVS and LODS are string transfer operations, XCHG exchanges operands, and INC is arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

> Data transfer instruction.
> Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Question sources

Computer Science model exam · 2015 · Question 42

Question 27 | Q00609

Which register is a user-accessible register?

A. DX

B. IR

C. MBR

D. MDR

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: DX is programmer-accessible.

COA21 explicitly identifies DX as programmer-visible and IR/MDR as conceptual internal registers. MBR is the memory-buffer name used for the same data-transfer role as MDR in COA11. The question asks which register is user-accessible in this architectural sense, so the x86 general-purpose register DX is the supported choice.

Why each choice is right or wrong

A - Correct: DX belongs to the programmer-visible x86 register set and is usable by ordinary instructions.

B - Incorrect: IR is the conceptual internal instruction register, rather than an ordinary programmer-addressable register.

C - Incorrect: MBR is the internal memory-buffer/data-transfer register, not a user-visible general-purpose register.

D - Incorrect: MDR is the conceptual internal memory-data register, not an ordinary user-accessible execution register.

Primary note proof

Study-note section COA21

DX is programmer-visible; IR/MDR are conceptual internal registers.

Study-note section COA11

A read places an address in MAR/AR, asserts read, waits for memory and captures data in MDR/MBR/DR.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

6.2.1 Connection of RAM and CPU
Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Question sources

Computer Science model exam · 2015 · Question 78

Question 28 | Q00614

Which register is found starting from Pentium processor?

A. EAX

B. EIP

C. R8

D. None

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: None was introduced with original Pentium.

Reading starting from as introduced with Pentium, COA21 rules out every named register. EAX and EIP already existed with the 80386, while R8 belongs to the later 64-bit extension rather than original Pentium. Therefore none is correct under the introduction-date reading. Simply asking which registers a Pentium contains would be a different question, with EAX and EIP both present.

Why each choice is right or wrong

A - Incorrect: EAX existed before Pentium in the 80386; it was not introduced starting with Pentium.

B - Incorrect: EIP also existed in the 80386, so Pentium was not its first generation.

C - Incorrect: R8 belongs to the later 64-bit extension, not the original Pentium register set.

D - Correct: None of the named registers was introduced with original Pentium under the starting-generation reading.

Primary note proof

Study-note section COA21

> EAX/EIP existed with the 80386, before Pentium; R8-R15 belong to the 64-bit extension, not original Pentium.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science model exam · 2015 · Question 83

Question 29 | Q00621

How many physical processor cores does the Intel Core i7-920 desktop processor have? Count cores, not hardware threads.

A. 5

B. 4

C. 6

D. 7

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: 4

The supplementary Intel desktop-processor comparison chart specifies four cores and eight threads for the i7-920. The answer is therefore four physical cores. The supplied note provides related background that Core i7 is a family with no single core count, but does not supply this model's number. Naming the model and consulting Intel supplies that missing fact without implying that every Core i7 has the same configuration.

Why each choice is right or wrong

A - Incorrect: Five does not match Intel's physical-core count for this model.

B - Correct: Intel lists four physical cores for the i7-920.

C - Incorrect: Six is not the specified model's physical-core count.

D - Incorrect: Seven is not this model's physical-core count.

Primary note proof

Study-note section COA-S01

> Intel's desktop comparison chart lists the Core i7-920 with four physical cores and eight hardware threads.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

> The following are important points about parallel processing:
> • A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
> • The system may have two or more ALUs and be able to execute two or more instructions at the same time
> • Also, the system may have two or more processors operating concurrently
> • Goal is to increase the throughput - the amount of processing that can be

accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science model exam · 2015 · Question 90

Question 30 | Q00628

A memory operand has effective address EA=A, where A is the address field stored directly in the instruction. Which addressing mode is used?

A. Direct addressing

B. Immediate addressing

C. Register addressing

D. None

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Direct addressing

In direct addressing, the instruction's address field is itself the effective address: EA=A, and the operand is M[A]. Immediate addressing uses A as a value rather than a memory address, while register addressing uses a register as the operand source. A transfer's register and memory endpoints alone do not determine its addressing mode, so the revised question supplies the missing effective-address relationship.

Why each choice is right or wrong

A - Correct: EA=A defines direct addressing.

B - Incorrect: Immediate addressing uses A as the operand value.

C - Incorrect: Register addressing uses a register operand rather than M[A].

D - Incorrect: Direct addressing matches the relationship.

Primary note proof

Study-note section COA18

Immediate uses value A; direct uses EA=A and operand M[A]; indirect uses EA=M[A] then operand M[EA].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
When the address field specifies a processor register, the instruction is said to be in the register mode.
Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

Data transfer instruction.
Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Question sources

Computer Science model exam · 2015 · Question 97

Question 31 | Q00630

One of the following is not a characteristic of SRAM

A. Fast

B. Expensive

C. Memory refreshing s

D. None

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Memory refreshing s

SRAM stores bits in flip-flop state; applying power is sufficient and no refresh is needed (PDF p.81). Fast and relatively expensive are expressly listed, making refreshing the exception.

Why each choice is right or wrong

A - Incorrect: Faster than DRAM is stated (p.81).

B - Incorrect: More expensive than DRAM is stated (p.81).

C - Correct: No need for refreshing is explicit (p.81), so this is the non-characteristic.

D - Incorrect: There is a non-characteristic, C.

Primary note proof

Study-note section COA24

> SRAM uses powered bistable storage: fast and costly per bit, without periodic refresh.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 81 | lines 6-16

> Static RAM:
> • a bit of data is stored using the state of a flip-flop.
> • Applying power is enough (no need for refreshing).
> • Retains value indefinitely, as long as it is kept powered.
> • Mostly uses to create cache memory of CPU
> • Faster and more expensive than DRAM.
> ROM
> ROM is used for storing programs that are permanently resident in the computer and for tables of constants that do not change in value once the production of the computer is

completed The ROM portion of main memory is needed for storing an initial program called bootstrap loader, which is to start the computer software operating when power is

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Question sources

Computer Science model exam · 2015 · Question 99

Question 32 | Q00632

Which one of the following is analog to digital converter?

A. ADC080X

B. DAC0830

C. 82C55

D. None

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: ADC080X is the analog-to-digital converter.

COA33 defines an ADC as producing a digital code from an analog input and specifically identifies ADC080x as an 8-bit successive-approximation ADC family. DAC0830 performs the reverse conversion. The 82C55 is a programmable parallel I/O interface, not an ADC. The component identifiers therefore resolve this question without inferring capabilities merely from their general interface role.

Why each choice is right or wrong

A - Correct: ADC080x is explicitly identified as the analog-to-digital converter family in the note.

B - Incorrect: DAC0830 converts digital codes into analog output, which is the reverse direction.

C - Incorrect: 82C55 provides programmable parallel I/O; the note explicitly distinguishes it from an ADC.

D - Incorrect: ADC080X is a valid analog-to-digital component, so the list is not missing an answer.

Primary note proof

Study-note section COA33

An ADC measures an analog input and produces a digital code; a DAC produces an analog output from a digital code.

Study-note section COA33

ADC080x parts are 8-bit successive-approximation analog-to-digital converters; DAC0830 is a digital-to-analog converter, while 82C55 denotes a programmable parallel I/O interface, not an ADC.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science model exam · 2015 · Question 101

Question 33 | Q00648

Which is NOT an Intel x86 16-bit general-purpose register?

A. AX

B. BP

C. SI

D. FLAGS

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: FLAGS

Intel’s general-purpose classification includes AX, BP and SI even though these registers have conventional special roles. FLAGS instead reports status conditions and is outside that group. The final option supplies an actual non-general-purpose register rather than incorrectly excluding SP merely because it tracks the stack. This preserves the register-class distinction without inventing a narrower meaning for multipurpose or misrepresenting Intel’s classification.

Why each choice is right or wrong

A - Incorrect: AX is a 16-bit general-purpose register.

B - Incorrect: BP is general-purpose despite its conventional base-pointer role.

C - Incorrect: SI is general-purpose despite its conventional source-index role.

D - Correct: FLAGS reports processor status rather than belonging to the general-purpose group.

Primary note proof

Study-note section COA21

> In Intel x86 notation, AX/BX/CX/DX, BP/SP and SI/DI are 16-bit general-purpose registers, with conventional special roles.

Study-note section COA21

> FLAGS/EFLAGS/RFLAGS report status; IP/EIP/RIP identifies instruction position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 69 | lines 15-23

> 5.15.4 Zero-Address Instruction
> A stack-organized computer does not use an address field for instructions ADD and MUL. The PUSH and POP instructions, however, need an address field to specify the operand that communicates with the stack. The following program shows how X = (A+B) * (C+D) will be written for a stack-organized computer (TOS stands for top-of-stack)

PUSH A TOS← A, PUSH B TOS ←B, ADD TOS ←(A + B),
PUSH C TOS ←C
PUSH D TOS ←D, ADD TOS ←(C + D) , MUL TOS ←(C + D) (A + B),
POP X M[X] ←TOS.

Question sources

Computer Science model exam · 2015 · Question 117

Question 34 | Q00689

Which of the following is a type of computer architecture?

A. Microarchitecture

B. Harvard Architecture

C. Von-Neumann Architecture

D. All of the mentioned

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: All listed categories are forms of computer design/architecture in the broad wording.

COA01 calls microarchitecture the internal processor design, while COA02 contrasts Harvard and Von Neumann memory organizations. The broad phrase type of computer architecture encompasses all three usages, although they describe different abstraction levels. The comprehensive answer is therefore all of the mentioned. No single named option covers the entire set of valid architectural categories.

Why each choice is right or wrong

A - Incorrect: Microarchitecture is a valid internal-design category, but this alone omits the two valid memory organizations.

B - Incorrect: Harvard is a valid architecture category, but selecting it alone excludes the other valid categories.

C - Incorrect: Von Neumann is valid, but the broad question also admits microarchitecture and Harvard organization.

D - Correct: All three names describe architectural design categories, though they operate at different abstraction levels.

Primary note proof

Study-note section COA01

Microarchitecture names the internal processor design.

Study-note section COA02

In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA02

Harvard separates instruction and data memories or paths.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 5 | lines 12-20

Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

Computer Science model exam 2 · Question 30

Question 35 | Q00690

Which one is not a combinational circuit?

A. Encoder

B. Decoder

C. Multiplexer

D. De-multiplexer

E. Register

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer E: Register

A register is a group of state-holding flip-flops (PDF p.35). Sequential circuits retain past information (p.15), whereas code/routing combinational circuits transform current inputs (pp.31,33-34). Thus register is the non-combinational choice.

Why each choice is right or wrong

A - Incorrect: Encoder maps the active input into a binary code (p.34); no stored state is part of this described mapping.

B - Incorrect: Decoder selects an output based on the input pattern (p.33).

C - Incorrect: Multiplexer is explicitly combinational (p.31).

D - Incorrect: Demultiplexer routes one input to a selected output (p.33), a present-input routing function.

E - Correct: Register contains storage flip-flops and retains state (p.35).

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

> • In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
> • However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
> • Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
> • Such circuits include storage elements that store the values of logic signals.
> • Flip-flops can store a one-bit data (information).

Computer Organization and Architecture module | PDF page 35 | lines 3-11

> 3.9 Registers
> • A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
> • A register is a digital circuit used within the CPU to store one or more bits of data.
> • An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
> • In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
> • Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 33 | lines 7-14

> 3.7 Decoders
> • A decoder is a combinational circuit with a number of output lines, only one of which

is selected at any time, depending on the pattern of input lines.
• In general, a decoder has n inputs and 2n
outputs.
• Decoders find many uses in digital computers.
• One example is address decoding.
• The other is binary-to- octal conversion.

Computer Organization and Architecture module | PDF page 34 | lines 3-8

3.8 Encoders
• An encoder is a digital circuit that performs the inverse operation of a decoder.
• An encoder has 2n
(or less) input lines and n output lines. The output lines generate
the binary code corresponding to the input value.
• An example of an encoder is the octal-to-binary encoder.

Computer Organization and Architecture module | PDF page 31 | lines 12-16

3.5 Multiplexers
• Multiplexer is a combinational circuit that receives binary information from one of 2n input data & directs to one output.
• The multiplexer connects multiple inputs to a single output.
• At any time, one of the inputs is selected to be passed to the output.

Computer Organization and Architecture module | PDF page 10 | lines 22-27

1.4 Combinational Circuits
• A combinational circuit is an interconnected set of gates whose output at any time is a function only of the input at that time. As with a single gate, the appearance of the input is followed almost immediately by the appearance of the output, with only gate delays. In general terms, a combinational circuit consists of n binary inputs and m binary outputs.

Question sources

Computer Science model exam 2 · Question 31

Question 36 | Q00691

In a _____ cache organization, cache permits data to be stored in any cache block.

A. Direct Mapped

B. Set Associativity

C. Fully Associative

D. All

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Fully Associative

Associative cache stores an address with each data item and searches by content (PDF p.87), rather than constraining a block to a single index. Set association retains index constraints (p.88). Fully associative is therefore the any-block placement organization.

Why each choice is right or wrong

A - Incorrect: Direct mapping has fixed index conflicts (p.88), so does not allow arbitrary block placement.

B - Incorrect: Set association allows multiple entries under the same index, not any block anywhere (p.88).

C - Correct: Address/content pairs searched associatively permit the unrestricted mapping described (p.87).

D - Incorrect: Direct and set mappings are restricted, so not all satisfy the claim.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Fully associative placement permits any line.

Study-note section COA25

> A W-way set-associative cache has S=L/W sets and set=block mod S; any of W ways in that set may hold the block.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 87 | lines 2-15

> Associative Mapping
> • An associative mapping uses an associative memory.
> • This memory is being accessed using its contents.
> • Associative memory is expensive compared to RAM
> • Each line of cache memory will accommodate the address (main memory) and the contents of that address from the main memory.
> • That is why this memory is also called Content Addressable Memory (CAM). It allows each block of main memory to be stored in the cache.
> A CPU address of 15 bits is places in the argument register and the associative memory us searched for a matching address
> • If the address is found, the corresponding 12bits data is read and sent to the CPU

• If not, the main memory is accessed for the word
If the cache is full, an address-data pair must be displaced to make room for a pair that is needed and not presently in the cache

Computer Organization and Architecture module | PDF page 88 | lines 2-13

Set Associative Mapping
That is the easy control of the direct mapping cache and the more flexible mapping of the fully associative cache.
In set associative mapping, each cache location can have more than one pair of tag + data items.
That is more than one pair of tag and data are residing at the same location of cache memory.
If one cache location is holding two pair of tag + data items, that is called 2-way set associative mapping.
The disadvantage of direct mapping is that two words with the same index in their address but with different tag values cannot reside in cache memory at the same time
Set-Associative Mapping is an improvement over the direct-mapping in that each word of cache can store two or more word of memory under the same index address

Question sources

Computer Science model exam 2 · Question 32

Question 37 | Q00692

Which one used to translate virtual addresses into physical addresses when we are implementing the concept of Virtual memory?

A. LRU

B. FIFO

C. Memory Management Unit

D. All

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Memory Management Unit (MMU).

COA27 states that the MMU translates virtual/logical addresses into physical addresses using page tables and often a TLB. LRU and FIFO instead choose replacement victims, as described in COA26. Neither replacement policy performs address translation. Consequently the MMU is the single component matching the requested virtual-memory translation role; the all alternative incorrectly includes two different mechanisms.

Why each choice is right or wrong

A - Incorrect: LRU chooses a least-recently-used replacement victim; it does not translate virtual addresses.

B - Incorrect: FIFO selects the earliest inserted replacement victim, rather than computing physical addresses.

C - Correct: The MMU performs virtual-to-physical address translation using the translation structures described in the note.

D - Incorrect: Only the MMU translates addresses; LRU and FIFO are replacement policies.

Primary note proof

Study-note section COA27

> The MMU translates virtual/logical addresses to physical addresses, using page tables and often a TLB of recent translations.

Study-note section COA26

> FIFO evicts the earliest inserted line; LRU evicts the least recently accessed line.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 90 | lines 9-23

> Concept of Virtual Memory
> A programmer can write a program which requires more memory space than the capacity of the main memory. Such a program is executed by virtual memory technique.
> The program is stored in the secondary memory. The memory management unit (MMU) transfers the currently needed part of the program from the secondary memory to the main memory for execution.
> This to and fro movement of instructions and data (parts of a program) between the main memory and the secondary memory is called Swapping.
> Address Space And Memory Space.
> Virtual address is the address used by the programmer and the set of such addresses is called the address space or virtual memory.
> An address in main memory is called a location or physical address. The set of such locations in main memory is called the memory space or physical memory.
> CPU generated logical address consisting of a logical page number plus the location within that page (x).

Computer Organization and Architecture module | PDF page 91 | lines 2-7

> It must be mapped onto an actual (physical) main memory address by the operating system using mapper.
> If the page is present in the main memory, CPU gets the required data from the main memory.
> If the mapper detects that the requested page is not present in main memory, a page fault occurs and the page must be read from secondary storage into a page frame in main memory.

Computer Organization and Architecture module | PDF page 88 | lines 14-22

> 6.3.2 Replacement Algorithms of Cache Memory
> Replacement algorithms are used when there are no available space in a cache in which to place a data. Four of the most common cache replacement algorithms are described below:
> Least Recently Used (LRU):
> The LRU algorithm selects for replacement the item that has been least recently used by the CPU.
> First-In-First-Out (FIFO):
> The FIFO algorithm selects for replacement the item that has been in the cache from the longest time.

Question sources

Computer Science model exam 2 · Question 33

Question 38 | Q00693

In a _____ bus, flow is only one-way from the CPU to memory.

A. Address Bus

B. Data Bus

C. Control Bus

D. A & C

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Address Bus

In the CPU-master model, CPU address registers identify the memory location (PDF pp.82-83), while data supports reads and writes (p.105) and control/status includes messages from both sides (pp.102-103). This makes A the specific one-way address path.

Why each choice is right or wrong

A - Correct: The CPU supplies the memory address (pp.82-83).

B - Incorrect: Read and write data travel in opposite directions (p.105).

C - Incorrect: Status reports return from I/O modules (p.103), so the whole control/status collection is not only CPU-outward.

D - Incorrect: C is not restricted to that one direction, so the conjunction fails.

Primary note proof

Study-note section COA12

> In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Study-note section COA12

Aggregate control signals include both CPU outputs and device inputs; both data and control buses can therefore be bidirectional under that scope.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

6.2.1 Connection of RAM and CPU
Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

• A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
• A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 103 | lines 2-9

Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
Address recognition:
just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Computer Organization and Architecture module | PDF page 102 | lines 20-24

I/O module must have the capability to engage in communication with the CPU and external device. Thus CPU communication involves
Command decoding: The I/O module accepts commands from the CPU carried on the control bus.
Data: data are exchanged between the CPU and the I/O module over data bus

Computer Organization and Architecture module | PDF page 83 | lines 2-7

During a “memory cycle‟ n bits of data are transferred between main memory and CPU. This transfer takes place over the processor bus, which has k address lines and n data lines. The CPU initiates a memory operation by loading the appropriate data into registers MDR and MAR, and setting either Read or Write memory control line to 1. When the required operation is completed the memory control circuitry sends Memory Function Completed (MFC) signal to CPU.

Question sources

Computer Science model exam 2 · Question 34

Question 39 | Q00694

In memory hierarchy, faster access then_____.

A. Greater cost per bit

B. Greater capacity

C. Smaller cost per bit

D. All

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer A: Greater cost per bit

The module states the speed/cost tradeoff directly: faster access has greater cost per bit (PDF p.79). Its separate capacity bullets associate larger capacity with lower cost and slower access, rather than faster access.

Why each choice is right or wrong

A - Correct: Exact stated speed/cost relationship (p.79).

B - Incorrect: Greater capacity is associated with slower access, not the stated faster tier (p.79).

C - Incorrect: Smaller cost per bit is associated with greater capacity, opposite the faster-access bullet (p.79).

D - Incorrect: B and C do not match the faster-access relationship.

Primary note proof

Study-note section COA23

> Faster levels are generally smaller and more expensive per bit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
> Figure 6.1 memory hierarchy

- Design constraints: How much? How fast? How expensive?
- Faster access time, greater cost per bit
- Greater capacity, smaller cost per bit,
- Greater capacity, slower access time.

Question sources

Computer Science model exam 2 · Question 35

Question 40 | Q00695

Which Shift register type used to act as a delay element?

A. Serial-In Serial-Out

B. Parallel-In Serial-Out

C. Serial-In Parallel-Out

D. All

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Serial-In Serial-Out

The shift register accepts/transfers data serially and moves each bit one position per clock, emitting the rightmost bit (PDF p.36). With serial input and serial output, an n-stage register delays the stream by n shifts. This derivation identifies SISO under the usual stream-delay meaning.

Why each choice is right or wrong

A - Correct: Serial input traverses storage stages to serial output, producing the clock-count delay (p.36).

B - Incorrect: Parallel input/serial output serializes a loaded word; it does not preserve the incoming serial stream in the described way.

C - Incorrect: Serial input/parallel output exposes the stored bits together rather than the delayed serial stream.

D - Incorrect: Not all listed I/O forms implement the same serial-stream delay.

Primary note proof

Study-note section COA13

SISO shift registers delay serial streams; SIPO collects a word and PISO serializes a parallel word.

Study-note section COA05

A flip-flop stores one bit while powered.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 36 | lines 2-9

Shift Register
• A shift register accepts and/or transfers information serially.
• Data are input only to the leftmost flip-flop.
• Shift registers can be used to interface to serial I/O devices.
• With each clock pulse, data are shifted to the right, one position, and the rightmost bit is transferred out.
• In addition, they can be used within the ALU to perform logical shift and rotate functions.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Question sources

Computer Science model exam 2 · Question 36

Question 41 | Q00697

Which one of them is a Universal Logic Gates?

A. NAND

B. Ex-OR

C. AND

D. NOT

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer A: NAND

Figure 1.2 on PDF p.7 gives NAND as NOT(A AND B). Tie its inputs to obtain NOT A; NAND followed by that inverter gives AND; applying NAND to inverted inputs gives OR, as can be checked against the same truth tables. Thus NAND alone can build the basic AND/OR/NOT set.

Why each choice is right or wrong

A - Correct: NAND supplies inversion by tying inputs and then AND/OR by composition (p.7, Figure 1.2).

B - Incorrect: XOR alone maps all-zero inputs to zero; it cannot supply the inverter required by Figure 1.2.

C - Incorrect: AND alone also preserves zero and cannot produce NOT using only AND gates.

D - Incorrect: NOT only complements one input; it cannot combine independent input variables into AND/OR.

Primary note proof

Study-note section COA04

> NAND and NOR are each universal: feeding the same input to both inputs constructs NOT; combine complemented gates to construct AND or OR.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Question sources

Computer Science model exam 2 · Question 38

Question 42 | Q00698

Which textbook architecture model stores instructions and data in a shared addressed memory system?

A. Microarchitecture

B. Harvard Architecture

C. Von-Neumann Architecture

D. System Design

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Von-Neumann Architecture

The Von Neumann model uses a shared addressed memory system for instructions and data. Harvard separates instruction and data memories or paths. Microarchitecture names an internal implementation, while system design is broader than the particular stored-program memory model requested. The revised stem asks about a defining memory relationship rather than claiming that all modern computers use one exclusive organization.

Why each choice is right or wrong

A - Incorrect: Microarchitecture describes internal processor implementation.

B - Incorrect: Harvard separates instruction and data memories or paths.

C - Correct: Von Neumann uses the described shared addressed memory model.

D - Incorrect: System design is broader than this named memory model.

Primary note proof

Study-note section COA02

> In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA02

> Harvard separates instruction and data memories or paths.

Study-note section COA01

> Microarchitecture names the internal processor design.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

> The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be

performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Question sources

Computer Science model exam 2 · Question 39

Question 43 | Q00893

In a design connecting a processor-side bus segment to an unlike peripheral bus segment, which hardware category provides the necessary bridge or interface logic?

A. SCSI

B. PCI

C. Bus bridge/controller logic

D. Multi bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Bus bridge/controller logic

Unlike bus segments need bridge or controller logic to adapt their interfaces and coordinate transfers. PCI names a peripheral expansion interconnect, while SCSI names a storage connection; neither alone identifies the adapting component. Multibus names a bus rather than the requested bridge category. The revised topology and component requirement make the interface-logic choice definite without assuming a particular processor bus or expansion standard.

Why each choice is right or wrong

A - Incorrect: SCSI is a storage-connection standard rather than a bridge component.

B - Incorrect: PCI names an expansion interconnect rather than the adapting logic.

C - Correct: Bridge/controller logic connects unlike bus segments.

D - Incorrect: Multibus names a bus rather than the required interface component.

Primary note proof

Study-note section COA32

> PCI is a peripheral expansion interconnect, not a disk protocol; connecting unlike bus segments requires suitable bridge/controller logic.

Study-note section COA32

> SATA is a storage interface; legacy IDE/PATA and SCSI represent other storage connections, while iSCSI carries SCSI commands over IP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

> 4.2 Bus and Memory Transfers
> A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

> When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
> • Whether to read or write
> • Device address
> • Starting address of memory block for data
> • Amount of data to be transferred The CPU carries on with other work.
> • Thus DMA controller steals the CPU‟s work of I/O operation.

• The DMA module transfers the entire block of data
• One word at a time, directly to or from memory, without going through CPU.
When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Question sources

Computer Science Exit Exam · 2018 · Question 6

Question 44 | Q00894

Combinational circuit differ from sequential circuit primarily because

A. Combinational circuit cannot be built using logic circuit

B. The output of combinational circuit depends only on the present input, not on past output.

C. .
Sequential circuit have no feedback while combinational circuit do

D. The combinational circuit requires clock signal, but sequential circuit do not

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: The output of combinational circuit depends only on the present input, not on past output.

Combinational output depends only on input at the current time (PDF p.10); sequential output also depends on the past and includes memory elements (p.15). This is the primary functional difference, independent of implementation speed.

Why each choice is right or wrong

A - Incorrect: The definition explicitly builds a combinational circuit from gates (p.10), contradicting A.

B - Correct: B restates the present-input-only definition (p.10).

C - Incorrect: Sequential circuits contain memory (p.15); reversing the feedback distinction does not match the combinational definition.

D - Incorrect: The module does not make a clock mandatory for combinational logic; it defines direct current-input dependence (p.10).

Primary note proof

Study-note section COA05

A combinational output depends on present inputs after propagation delay.

Study-note section COA05

A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Study-note section COA05

Clocked sequential circuits update at designated clock events; combinational logic does not inherently require a clock.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 10 | lines 22-27

1.4 Combinational Circuits
• A combinational circuit is an interconnected set of gates whose output at any time is a function only of the input at that time. As with a single gate, the appearance of the input is followed almost immediately by the appearance of the output, with only gate delays. In general terms, a combinational circuit consists of n binary inputs and m binary outputs.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Question sources

Computer Science Exit Exam · 2018 · Question 7

Question 45 | Q00909

Which one of the following is the first operation or step in CPU instruction cycle?

A. Executing the instruction

B. Handling interrupt

C. .
Decoding the instruction

D. Fetching the instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Fetching the instruction

The listed cycle begins with fetching an instruction, then decoding, possible indirect-address access, and execution (PDF p.55). In the fetch micro-operations PC first supplies AR and memory then supplies IR.

Why each choice is right or wrong

A - Incorrect: Execution follows fetch and decode, rather than initiating the cycle (p.55).

B - Incorrect: Interrupt handling is not the first phase in the module's listed ordinary instruction cycle.

C - Incorrect: Decoding is the second listed phase (p.55).

D - Correct: Fetching is explicitly phase 1, with register transfers shown on p.55.

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

> 1. Fetch an instruction from memory
> 2. Decode the instruction
> 3. Read the effective address from memory if the instruction has an indirect address
> 4. Execute the instruction
> Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
> T0: AR ← PC
> Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented

T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1,
INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science Exit Exam · 2018 · Question 22

Question 46 | Q00913

Which one of the following is a computer structural component that provides for communication among CPU, main memory and I/O?

A. System unit

B. Computer network

C. CPU

D. System interconnection

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: System interconnection

The structural-component list defines system interconnection as the mechanism providing communication among CPU, main memory and I/O (PDF p.6). This definition exactly matches the requested role.

Why each choice is right or wrong

A - Incorrect: “System unit” is not the named communication component in the list on p.6.

B - Incorrect: A network is not the internal mechanism named in this structural definition.

C - Incorrect: CPU is one of the three communicating parties, not the interconnection itself (p.6).

D - Correct: The definition of system interconnection matches the entire stem (p.6).

Primary note proof

Study-note section COA01

CPU, main memory, I/O and their interconnection are the major structural parts.

Study-note section COA-S02

A bus is a set of common signal lines used to transfer binary information among connected units.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Computer Science Exit Exam · 2018 · Question 26

Question 47 | Q00918

is a data transfer method in computer organization and architecture that allows an I/O device to transfer data directly to or from memory without the involvement of the CPU.

A. Asynchronous data transfer

B. Direct Memory Access (DMA)

C. .
Interrupt-driven transfer

D. Programmed I/O transfer

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Direct Memory Access (DMA)

DMA sends the block directly to/from memory and keeps CPU involvement to setup and completion (PDF pp.107-108). Asynchrony describes timing, whereas interrupt/programmed I/O still make the CPU transfer data.

Why each choice is right or wrong

A - Incorrect: Asynchronous transfer describes independent clocks (p.101), not a bypass path.

B - Correct: DMA words go directly to or from memory without traversing CPU (p.108).

C - Incorrect: The CPU executes transfers under interrupt I/O (p.106).

D - Incorrect: Programmed I/O has CPU status checks and data transfer (p.104).

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer’s motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

> When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
> • Whether to read or write
> • Device address
> • Starting address of memory block for data
> • Amount of data to be transferred The CPU carries on with other work.
> • Thus DMA controller steals the CPU‟s work of I/O operation.
> • The DMA module transfers the entire block of data
> • One word at a time, directly to or from memory, without going through CPU.
> When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 101 | lines 2-9

> Asynchronous data transfer
> The internal operation in digital system is synchronized by means of clock pulse supplied by a common pulse generator. Two units such as cpu and an I/O are designed independently of each other, if the register in the interface share a common clock with the cpu registers, the transfer between the two units is called to be synchronous.in most case the internal timing in each unit is independent from the other in that each uses its own private clock for internal

register.in such cases the two units are said to be asynchronous to each other. This approach is widely used in most computers systems.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Question sources

Computer Science Exit Exam · 2018 · Question 31

Question 48 | Q01178

Which instruction class tests a data word or computation status and sets condition information, often without retaining an arithmetic result?

A. Test instructions

B. Branch instructions

C. All of the above

D. Data-transfer instructions

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Test instructions

TEST/CMP-style instructions examine values and update condition information, often discarding an arithmetic result. Branch instructions use control-flow conditions to choose a target; data-transfer instructions copy values. Those are different roles, so the aggregate alternative cannot identify the testing class. The cleaned question retains the recognizable, image-confirmed test-instruction concept while removing displaced OCR debris.

Why each choice is right or wrong

A - Correct: Test instructions examine values and set condition information.

B - Incorrect: Branches choose control-flow targets rather than naming the class that establishes the test condition.

C - Incorrect: Branch and data-transfer instructions do not both have the specified test role, so the aggregate is false.

D - Incorrect: Data-transfer instructions copy values rather than performing the requested condition-setting test.

Primary note proof

Study-note section COA19

> TEST/CMP examine values and set condition information, often without retaining the arithmetic result.

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Study-note section COA19

> Data-transfer instructions copy values.

Study-note section COA19

> Arithmetic/logic instructions compute values.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

1. Fetch an instruction from memory
2. Decode the instruction
3. Read the effective address from memory if the instruction has an indirect address
4. Execute the instruction
Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 51 | lines 14-22

Instruction Set Completeness
A computer should have a set of instructions so that the user can construct machine language program to evaluate any function that is known to be computable. The set of instructions are said to be complete if the computer includes a sufficient number of instructions in each of the following categories:
1. Arithmetic, Logical and Shift Instructions.
2. Instructions for moving information to and from memory and processor registers.
3. Program control instructions together with instructions that check status conditions.
4. Input and Output instructions.

Question sources

Computer Science Exit Exam practice questions · Question 39

Question 49 | Q01179

Branch instructions are used to branch to a different set of instructions depending on the decision made.

A. True

B. False

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: True

A branch changes the ordinary sequential instruction path to the branch address. The module explicitly identifies the actual branch address in a branch instruction (PDF p.72), and visually inspected Table 5-2 (p.52) shows both unconditional branch and conditional skip operations. Interpret “depending on the decision” as conditional branching; it is not true that every branch must be conditional.

Why each choice is right or wrong

A - Correct: True under the conditional-branch interpretation: the address field supplies the alternate target (p.72).

B - Incorrect: False would deny that conditional transfers select another instruction sequence; the table’s conditional skip examples illustrate such decisions (p.52).

Primary note proof

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

> Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
> Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

1. Fetch an instruction from memory
2. Decode the instruction
3. Read the effective address from memory if the instruction has an indirect address
4. Execute the instruction
Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 52 | lines 2-6

Table 5-2: Basic computer instructions
Arithmetic, logical and shift instructions provide computational capabilities for processing the type of data that the user may wish to employ. The bulk of the binary information in a digital computer is stored in memory, but all computations are done in processor registers. Therefore, the user must have the capability of moving information between these two units

Question sources

Computer Science Exit Exam practice questions · Question 40

Question 50 | Q01183

Which of the following is not considered as a peripheral device?

A. CPU

B. Keyboard

C. Monitor

D. All of the above

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: CPU

Peripherals are external input/output devices such as keyboard and display (PDF p.99). CPU is the central processor structural component (p.6), so it is the only offered non-peripheral.

Why each choice is right or wrong

A - Correct: CPU is the processor, distinct from the external I/O devices (p.6).

B - Incorrect: Keyboard is explicitly a peripheral (p.99).

C - Incorrect: Display/video monitor is explicitly peripheral (p.99).

D - Incorrect: Keyboard and monitor are peripherals, so not all choices satisfy the negative stem.

Primary note proof

Study-note section COA29

> A peripheral performs input, output or storage outside the CPU's execution core; CPU itself is not a peripheral.

Study-note section COA29

> Input units encode external information and output units present processed results.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 99 | lines 3-7

> 7.1 Peripheral (External) Devices
> Input or output devices attached to the computer are called peripherals. Among the most known peripherals some of them are, keyboard display unit and printers.
> Video monitors are the most commonly used peripheral. They consists keyboard as the input device and display unit as an output devices.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science model exam 2 · Question 4

Question 51 | Q01187

Which one of the following statements is not true about the application of micro-operations?

A. Selective set operation is implemented using OR micro-operation

B. Selective clear operation clears to zero the bit in the first operand only where there are corresponding 1's in the logical operand

C. Clear operation is implemented using Exclusive-OR Micro-operation.

D. None of the above

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: None of the above

Selective set uses A OR B and selective clear zeroes A where B has ones (PDF p.44). For the unqualified clear operation, XOR a register with itself: each bit x XOR x=0, as shown by the equal-input rows of Figure 1.2 on PDF p.7. Thus A-C can all be true and D answers the negative stem.

Why each choice is right or wrong

A - Incorrect: The OR expression and example for selective set are explicit (p.44).

B - Incorrect: The selective-clear definition exactly states zeroing A at B=1 positions (p.44).

C - Incorrect: XOR self-clear is valid because 0 XOR 0 and 1 XOR 1 both equal zero (p.7, visually checked truth table).

D - Correct: None of A-C is false under self-XOR clearing, so D is the exception response.

Primary note proof

Study-note section COA13

> To set selected bits use A OR mask; to clear them use A AND NOT(mask); to toggle use A XOR mask; to retain only selected bits use A AND mask.

Study-note section COA13

> A XOR A clears the entire register.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 44 | lines 16-31

Selective-set: The selective-set operation sets to 1 the bits in register A where there are corresponding 1's in register B. It does not affect bit positions that have 0's in B.
e.g 1010 A Before
1100 B (logic operand)
1110 A after, the corresponding micro operation is A←A V B
Selective-complement: The selective-complement operation complements bits in A where there are corresponding 1's in B. It does not affect bit positions that have 0's in B.
e.g 1010 A before
1100 B (logic operand)
0110 A after , the corresponding logic micro operation A←A B
Selective-clear: The selective-clear operation clears to 0 the bits in A only where there are corresponding 1's in B.
e.g 1010 A before
1100 B (logic operand)
0010 A after
The corresponding logic micro operation is A← A ∧¯

Computer Organization and Architecture module | PDF page 7 | lines 3-9

Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 38 | lines 8-12

Digital modules are best defined by the registers they contain and the operations that are performed on the data stored in them. The operations executed on data stored in registers are called micro operations. A micro operation is an elementary operation performed on the information stored in one or more registers.
Examples of micro operation are shift, count, clear, and load.

Computer Organization and Architecture module | PDF page 75 | lines 5-13

b. Logical and Bit manipulation Instructions
Logical instructions perform binary operations on strings of bits stored in registers. They are useful for manipulating individual bits or a group of bits that represent binary-coded information. The logical instructions consider each bit of the operands separately and treat it as a Boolean variable. By proper application of the logical instructions it is possible to change bit values, to clear a group of bits or to insert new bit values into the operands stored in registers or memory words. Some logical and bit manipulation instructions are listed in Table 5-8 as follows.
Table 5-8 Some logical and bit manipulation instructions

Question sources

Computer Science model exam 2 · Question 8

Question 52 | Q01190

In textbook discussions of richer CISC instructions, which term names the distance between high-level language constructs and machine operations that those instructions may reduce?

A. Time delay

B. Semantic gap

C. Cost

D. All of the mentioned

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer B: Semantic gap

The semantic gap is the distance between high-level constructs and machine operations. Richer instructions can reduce that gap by expressing elaborate operations directly. Time and cost are separate design concerns, and neither RISC nor CISC guarantees reductions in them across implementations and workloads. The revised question ties the named gap to the supported richer-instruction discussion instead of asserting one universal purpose shared by both approaches.

Why each choice is right or wrong

A - Incorrect: Time delay is a performance measure rather than this conceptual distance.

B - Correct: Semantic gap names the high-level-to-machine-operation distance.

C - Incorrect: Cost is an implementation measure rather than this gap.

D - Incorrect: The question asks for one specific conceptual term.

Primary note proof

Study-note section COA20

> The semantic gap is the distance between high-level constructs and machine operations; reducing it is associated with richer instructions, not a universal shared objective of both design approaches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions
> • All operations done within the register of the CPU

• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Question sources

Computer Science model exam 2 · Question 11

Question 53 | Q01248

Which register identifies the next instruction to fetch, before its address is copied into the address register for a memory access?

A. Accumulator

B. Instruction register

C. Address register

D. Program counter

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Program counter

The program counter identifies the next instruction to fetch. During fetch, that address can be copied into MAR so memory can perform the access, but MAR's general role is the address of the current memory operation. IR holds the fetched instruction itself and the accumulator holds operands or results. Specifying next instruction and the point before copying separates PC from the later memory-access address.

Why each choice is right or wrong

A - Incorrect: The accumulator holds operands or results.

B - Incorrect: IR holds the fetched instruction itself.

C - Incorrect: MAR holds the current memory-access address after it is loaded.

D - Correct: PC identifies the next instruction to fetch.

Primary note proof

Study-note section COA14

> PC identifies the next instruction to fetch.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

> Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science model exam 2 · Question 69

Question 54 | Q01250

Which of the following logic gates has an output value that is identical to its input?

A. NAND

B. NOR

C. Exclusive-OR

D. Buffer

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer D: Buffer

The enabled three-state buffer has output Y=A when control C=1 (PDF p.41). Therefore buffer preserves the input logic value. Figure 1.2 on p.7 gives the other gates' differing two-input truth tables.

Why each choice is right or wrong

A - Incorrect: NAND complements AND; it is not an identity function (p.7).

B - Incorrect: NOR complements OR; it is not an identity function (p.7).

C - Incorrect: XOR depends on both inputs; it equals one input only under an extra fixed-zero assumption (p.7).

D - Correct: An enabled buffer explicitly gives Y=A (p.41).

Primary note proof

Study-note section COA03

> An enabled buffer copies its input.

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 41 | lines 12-18

> The bus system can be constructed with three-state gates instead of multiplexers. A three-state gate is a digital circuit that exhibits three states. Two of the states are signals equivalent to logic 1 and 0 as in a conventional gate. The third state is a high- impedance state. The high-impedance state behaves like an open circuit, which means that the output is disconnected and does not have logic significance.
> Output Y=A if C=1
> High Impedance if C=0

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Question sources

Computer Science model exam 2 · Question 71

Question 55 | Q01280

From the labels RISC and CISC alone, which conclusion about processor power efficiency is justified without implementation or workload information?

A. RISC

B. Neither label alone determines power efficiency

C. IANA

D. CISC

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer B: Neither label alone determines power efficiency

RISC and CISC describe instruction-set design tendencies, not universal power rankings. Power efficiency depends on implementation and the work being performed, so neither label alone establishes which processor is more efficient. The option states that supported conclusion. RISC and CISC remain unjustified blanket selections, while IANA does not supply the processor implementation or workload needed to resolve the comparison.

Why each choice is right or wrong

A - Incorrect: RISC does not guarantee greater power efficiency.

B - Correct: The note rejects a guaranteed power ranking from these labels alone.

C - Incorrect: This does not provide a relevant processor implementation for the comparison.

D - Incorrect: CISC does not guarantee greater power efficiency.

Primary note proof

Study-note section COA20

> Neither label guarantees lower power, cost or execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions
> • All operations done within the register of the CPU
> • Fixed length, easily decoded instruction format
> • Single cycle instruction execution
> • Hard-wired rather than micro-programmed control
> • A relatively large number of registers in the processor

• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Computer Organization and Architecture module | PDF page 118 | lines 3-19

A RISC (Reduced Instruction Set Computer) processor pipeline operates in much the same way, although the stages in the pipeline are different. While different processors have different numbers of steps, they have basically variations of these five steps:
1. fetch instructions from memory
2. read registers and decode the instruction
3. execute the instruction or calculate an address
4. access an operand in data memory
5. write the result into a register
The length of the pipeline is dependent on the length of the longest step. Because RISC instructions are simpler than those used in pre-RISC processors (now called CISC, or Complex Instruction Set Computer), they are more conducive to pipelining. While CISC instructions varied in length, RISC instructions are all the same length and can be fetched in a single operation. Ideally, each of the stages in a RISC processor pipeline should take 1 clock cycle so that the processor finishes an instruction each clock cycle and averages one cycle per instruction (CPI). Among the characteristics attributed to RISC is its ability to use an efficient instruction pipeline.

Question sources

Computer Science model exam 2 · Question 101

Question 56 | Q01304

Which of the following is not a type of memory used in an operating system?

A. Cache

B. RAM

C. ROM

D. Ethernet

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Ethernet is a network connection, not a memory type.

COA23 describes cache as copying memory blocks near the CPU, and COA24 describes RAM technologies plus nonvolatile ROM. COA32 explicitly classifies Ethernet as connecting network devices rather than memory. Thus cache, RAM and ROM are memory-related choices, while Ethernet belongs to communications. The phrase used in an operating system does not convert a network technology into storage hardware.

Why each choice is right or wrong

A - Incorrect: Cache stores copies of memory blocks and is a genuine level of the memory hierarchy.

B - Incorrect: RAM is main-memory storage, not a communication technology excluded from the memory category.

C - Incorrect: ROM is nonvolatile memory even though its normal access and modification rules differ from RAM.

D - Correct: Ethernet connects network devices and is explicitly distinguished from memory in the note.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA24

> ROM and flash retain contents without power; erasable/programmed variants mean “ROM can never be modified” is too broad.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

> 6.2 Main Memory
> The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 81 | lines 6-16

> Static RAM:
> • a bit of data is stored using the state of a flip-flop.
> • Applying power is enough (no need for refreshing).

• Retains value indefinitely, as long as it is kept powered.
• Mostly uses to create cache memory of CPU
• Faster and more expensive than DRAM.
ROM
ROM is used for storing programs that are permanently resident in the computer and for tables of constants that do not change in value once the production of the computer is completed The ROM portion of main memory is needed for storing an initial program called bootstrap loader, which is to start the computer software operating when power is

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Question sources

Operating Systems practice questions · Question 22

Question 57 | Q01318

Which of the following is not a type of storage device used in an operating system?

A. Hard disk drive (HDD)

B. Solid-state drive (SSD)

C. Tape drive

D. Ethernet cable

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Ethernet cable is a communication medium.

COA28 describes HDD magnetic storage, SSD nonvolatile semiconductor storage and tape for sequential archives. It then explicitly distinguishes an Ethernet cable as a communication medium rather than a disk or RAM. The question asks for the item outside the storage-device category, so the cable is the supported exclusion regardless of which filesystem or OS uses the other devices.

Why each choice is right or wrong

A - Incorrect: An HDD stores persistent magnetic patterns in addressed blocks, so it is a storage device.

B - Incorrect: An SSD stores data in nonvolatile semiconductor storage, satisfying the storage-device category.

C - Incorrect: A tape drive provides sequential storage suitable for streaming and archives, so it is not excluded.

D - Correct: An Ethernet cable carries communication signals rather than functioning as a persistent storage device.

Primary note proof

Study-note section COA28

> An HDD stores magnetic patterns on tracks divided into sectors; reads/writes access addressed blocks while platters rotate.

Study-note section COA28

> SSDs use nonvolatile semiconductor storage without mechanical seek.

Study-note section COA28

Optical media use lasers; tape is suited to sequential streaming and archives.

Study-note section COA28

An Ethernet cable is a communication medium, not a disk or RAM.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Computer Organization and Architecture module | PDF page 98 | lines 7-16

6.4.2 Magnetic Tape
• Magnetic Tape is a plastic ribbon which is usually ½ inch or ¼ inch wide & 50 to 2400 feet long.
• It is coated with iron-oxide material.
• It is similar to the tape of audio cassettes of tape recorders. Data is stored as binary digits.
• Data is accessed sequentially so searching becomes difficult.
• Tape systems use the same reading and recording techniques as disk systems..
• Magnetic tape was the first kind of secondary memory.It is still widely used as the lowest-cost, slowest-speed member of the memory hierarchy.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.

• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Operating Systems practice questions · Question 36

Question 58 | Q01465

Which generation of computers used vacuum tubes as their primary electronic component?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer A: First generation: vacuum tubes.

Under the familiar generation convention in COA38, vacuum tubes identify the first generation. Transistors belong to the second, integrated circuits to the third, and microprocessor/VLSI-era personal computers to the fourth. This question asks for the defining electronic component in that convention, so the mapping provides a direct answer without needing exact boundary dates for every historical machine.

Why each choice is right or wrong

A - Correct: The first generation is the conventional vacuum-tube era specified in the complete note.

B - Incorrect: The second generation is associated with transistors rather than vacuum tubes as its defining component.

C - Incorrect: The third generation is associated with integrated circuits rather than the vacuum-tube era.

D - Incorrect: The fourth generation is associated with microprocessors/VLSI rather than vacuum tubes as its primary component.

Primary note proof

Study-note section COA38

The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Operating Systems practice questions · Question 13

Question 59 | Q01466

Which generation of computers utilized transistors for electronic circuitry?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer B: Second generation: transistors.

COA38 maps the second generation to transistor-based circuitry. The surrounding categories distinguish the first vacuum-tube era, third integrated-circuit era and fourth microprocessor/VLSI era. Within this conventional classification, the transistor clue therefore selects the second generation. The answer concerns the defining technology, rather than claiming transistors ceased to exist when integrated circuits appeared.

Why each choice is right or wrong

A - Incorrect: The first generation is conventionally identified with vacuum tubes, not transistor-based circuitry.

B - Correct: The note explicitly associates transistor circuitry with the second generation of computers.

C - Incorrect: Integrated circuits define the third-generation category, rather than the discrete-transistor era requested here.

D - Incorrect: Microprocessor/VLSI-era computers define the fourth category, rather than the transistor-generation convention.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

> DYNAMIC RAM (DRAM)
> is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
> STATIC RAM (SRAM)
> In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 30 | lines 3-19

> 3.1 Integrated Circuits
> • Integrated circuit (IC) is the basic building block of digital circuits.
> • An integrated circuit is a small silicon semiconductor crystal, called a chip.
> • The various gates are interconnected inside the chip to form the required circuit.
> • As the technology of ICs has improved, the number of gates that can be put in a single chip has increased.
> ✓ Small-scale integration (SSI) devices contain several (usually less than 10) independent gates in a single package.
> ✓ Medium-scale integration (MSI) devices contain approximately 10 to 200 gates in a single package.
> E.g To form decoders, adders, and registers.
> ✓ Large-scale integration (LSI) devices contain between 200 and a few thousands gates in a single package.
> Eg. processors, memory chips, and programmable modules.
> ✓ Very-large-scale integration (VLSI) devices contain thousands of gates in a single package.
> E.g large memory arrays and complex microcomputer chips.

Question sources

Operating Systems practice questions · Question 14

Question 60 | Q01467

Under the course's conventional generation summary, around the transition into which generation did high-level programming languages become prominent?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer B: Second Generation

The course associates high-level languages becoming prominent with the transition into the second generation. That is a conventional historical summary, rather than a claim that every language was first introduced on one sharply defined boundary. The revised verb preserves the generation-history question while avoiding an unsupported universal introduction era. The other options refer to different stages in the familiar circuitry-based chronology.

Why each choice is right or wrong

A - Incorrect: The note instead places prominence around the transition into the second era.

B - Correct: This matches the explicitly stated historical convention.

C - Incorrect: The third era is associated with integrated circuits rather than this transition.

D - Incorrect: The fourth era is associated with microprocessor/VLSI personal computers.

Primary note proof

Study-note section COA38

> High-level languages became prominent around the transition into the second; introduction dates do not fit every generation boundary.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions

• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Operating Systems practice questions · Question 15

Question 61 | Q01468

Which generation of computers introduced the concept of stored program architecture?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer A: First generation.

COA38 directly places stored-program machines in the first era. The stored-program clue is different from the transistor, integrated-circuit and microprocessor clues defining later generations. Under the historical convention supplied by this note, the concept therefore belongs to the first generation, even though later computers also retained stored-program operation.

Why each choice is right or wrong

A - Correct: The note explicitly places the rise of stored-program machines in the first generation.

B - Incorrect: The second generation retained stored programs but is not where the note places their introduction.

C - Incorrect: The third generation is the integrated-circuit era, later than the stated stored-program introduction.

D - Incorrect: The fourth generation is the microprocessor era, much later than the stated first-era development.

Primary note proof

Study-note section COA38

> Stored-program machines arose in the first era.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

> The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Question sources

Operating Systems practice questions · Question 16

Question 62 | Q01469

Which generation of computers saw the development of integrated circuits?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer C: Third generation: integrated circuits.

COA38 associates the third generation with integrated circuits. The first is identified by vacuum tubes, the second by transistors and the fourth by microprocessor/VLSI-era personal computers. Reading development in the question as the conventional adoption/defining-technology category gives third generation; it does not require assuming that every invention date perfectly matches textbook boundaries.

Why each choice is right or wrong

A - Incorrect: The first-generation convention identifies vacuum tubes rather than integrated circuits as the defining technology.

B - Incorrect: The second-generation convention identifies transistor circuitry, preceding the integrated-circuit generation category.

C - Correct: Integrated circuits are the defining third-generation technology in the supplied historical convention.

D - Incorrect: The fourth generation is the later microprocessor/VLSI category rather than the initial integrated-circuit category.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 30 | lines 3-19

> 3.1 Integrated Circuits
> • Integrated circuit (IC) is the basic building block of digital circuits.
> • An integrated circuit is a small silicon semiconductor crystal, called a chip.
> • The various gates are interconnected inside the chip to form the required circuit.
> • As the technology of ICs has improved, the number of gates that can be put in a single chip has increased.
> ✓ Small-scale integration (SSI) devices contain several (usually less than 10) independent gates in a single package.
> ✓ Medium-scale integration (MSI) devices contain approximately 10 to 200 gates in a single package.
> E.g To form decoders, adders, and registers.
> ✓ Large-scale integration (LSI) devices contain between 200 and a few thousands gates in a single package.
> Eg. processors, memory chips, and programmable modules.
> ✓ Very-large-scale integration (VLSI) devices contain thousands of gates in a single package.
> E.g large memory arrays and complex microcomputer chips.

Question sources

Operating Systems practice questions · Question 17

Question 63 | Q01470

A historical computer uses magnetic-core memory and transistor-based circuitry rather than vacuum tubes or integrated circuits. Under the conventional taxonomy, which generation describes its circuitry?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer B: Second Generation

Magnetic-core memory alone does not identify a generation because it spanned more than one era. The added circuitry condition supplies the distinguishing feature: the conventional taxonomy associates transistor-based machines with the second generation. Vacuum tubes identify the first era, integrated circuits the third and microprocessor/VLSI personal computers the fourth. This version retains the magnetic-core context and states which technology determines the classification.

Why each choice is right or wrong

A - Incorrect: First-generation circuitry uses the vacuum-tube convention.

B - Correct: Transistor circuitry identifies the second generation.

C - Incorrect: Third-generation circuitry uses the integrated-circuit convention.

D - Incorrect: The fourth era is associated with microprocessor/VLSI systems.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Study-note section COA38

> Magnetic-core memory spanned more than one era, so “witnessed use” cannot identify exactly one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

6.2 Main Memory
The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Operating Systems practice questions · Question 18

Question 64 | Q01472

Which generation conventionally introduced integrated-circuit circuitry, contributing to improved speed and performance over earlier discrete-transistor systems?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer C: Third Generation

Several generations improved speed, so improvement alone is not a unique historical marker. The revised question identifies integrated circuits as the distinguishing technology. Under the stated convention, integrated circuits identify the third generation, following transistor-based second-generation systems. Performance improvement remains a contextual trend rather than a guarantee that every individual later machine outperformed every earlier model.

Why each choice is right or wrong

A - Incorrect: The first era is associated with vacuum tubes.

B - Incorrect: The second era is associated with discrete transistors.

C - Correct: The third era is associated with integrated circuits.

D - Incorrect: The fourth era is associated with microprocessor/VLSI systems.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Study-note section COA38

> Integration generally reduced individual component dimensions while capacity, density and processor performance grew; these are trends rather than guarantees for every model.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions
> • All operations done within the register of the CPU
> • Fixed length, easily decoded instruction format
> • Single cycle instruction execution
> • Hard-wired rather than micro-programmed control
> • A relatively large number of registers in the processor
> • Use overlapped register windows to speed up procedure call and return
> • Efficient instruction pipeline
> • Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of

memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Question sources

Operating Systems practice questions · Question 20

Question 65 | Q01473

Which generation of computers introduced the use of microprocessors?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer D: Fourth generation: microprocessors.

The conventional mapping in COA38 identifies the fourth generation with microprocessor/VLSI-era computers. Vacuum tubes, transistors and integrated circuits identify the preceding first, second and third categories respectively. Thus the microprocessor clue directly selects the fourth generation under the note's convention; the question does not ask for an exact launch date or individual processor model.

Why each choice is right or wrong

A - Incorrect: Vacuum tubes define the first-generation convention, rather than the later microprocessor era.

B - Incorrect: The transistor-based second-generation category precedes the note's microprocessor/VLSI category.

C - Incorrect: Third generation denotes integrated-circuit computers; microprocessors identify the subsequent conventional category.

D - Correct: The complete note explicitly associates microprocessor/VLSI-era computers with the fourth generation.

Primary note proof

Study-note section COA38

The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Operating Systems practice questions · Question 21

Question 66 | Q01474

Which generation of computers is characterized by the development of personal computers?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer D: Fourth generation: personal computers.

COA38 groups personal computers with the fourth-generation microprocessor/VLSI era. The first three alternatives instead correspond to vacuum tubes, transistors and integrated circuits. The wording characterized by asks for that conventional defining association, so fourth generation is supported without making a universal claim that no personal or individual-use machine could exist in any earlier period.

Why each choice is right or wrong

A - Incorrect: The first era is characterized by vacuum tubes in this convention, not microprocessor-based personal computers.

B - Incorrect: The second era is the transistor category rather than the personal-computer era identified here.

C - Incorrect: The third era is associated with integrated circuits rather than the note's personal-computer category.

D - Correct: The note places microprocessor/VLSI-era personal computers in the fourth-generation historical convention.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Operating Systems practice questions · Question 22

Question 67 | Q01491

A stack organized computer uses instruction of

A. Zero addressing

B. Two addressing

C. Index addressing

D. Indirect addressing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Zero addressing

For stack arithmetic the operands are implicit on top of stack, so ADD and MUL require no address field (PDF p.69). This is the zero-address organization. PUSH/POP still use explicit addresses; the answer concerns arithmetic operation instructions.

Why each choice is right or wrong

A - Correct: Zero-address ADD/MUL is explicit (p.69).

B - Incorrect: Two-address instructions instead explicitly name register/memory operands; stack arithmetic does not (pp.67-69).

C - Incorrect: Index addressing is an effective-address mechanism, not the stack operand-count format.

D - Incorrect: Indirect addressing locates an address through memory (p.72); it is not the stack arithmetic format.

Primary note proof

Study-note section COA16

> Zero-address stack ADD uses two implicit top-of-stack operands.

Study-note section COA17

> A stack is last-in, first-out.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 69 | lines 15-23

> 5.15.4 Zero-Address Instruction
> A stack-organized computer does not use an address field for instructions ADD and MUL.
> The PUSH and POP instructions, however, need an address field to specify the operand that communicates with the stack. The following program shows how X = (A+B) * (C+D) will be written for a stack-organized computer (TOS stands for top-of-stack)
> PUSH A TOS← A, PUSH B TOS ←B, ADD TOS ←(A + B),

PUSH C TOS ←C
PUSH D TOS ←D, ADD TOS ←(C + D) , MUL TOS ←(C + D) (A + B),
POP X M[X] ←TOS.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
When the address field specifies a processor register, the instruction is said to be in the register mode.
Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Question sources

Mock exam 1 · Question 7

Question 68 | Q01492

An n-bit microprocessor has -

A. No answer is correct

B. n-bit program counter

C. n-bit address register

D. n-bit instruction register

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: None of the named register widths necessarily follows.

COA38 explicitly states that an n-bit processor label does not force its PC, IR or address register to be n bits. Those widths concern different architectural quantities. Since the stem supplies only the processor-width label and no specific design, none of the three claimed register widths is guaranteed. The no-answer-is-correct alternative is therefore the supported response.

Why each choice is right or wrong

A - Correct: None of the three specific register-width claims is implied by the processor-width label alone.

B - Incorrect: Program-counter width is not forced to equal the processor's n-bit label in every architecture.

C - Incorrect: Address-register width can differ from processor word width; n bits is not guaranteed.

D - Incorrect: Instruction-register width depends on instruction encoding rather than necessarily matching the processor-width label.

Primary note proof

Study-note section COA38

> An n-bit processor label does not force its PC, IR or address register to be n bits.

Study-note section COA12

> Word width, data-bus width, instruction width and address width need not match.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

> 3.9 Registers
> • A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
> • A register is a digital circuit used within the CPU to store one or more bits of data.
> • An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
> • In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
> • Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

> The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Question sources

Mock exam 1 · Question 8

Question 69 | Q01493

In which of the following term the performance of cache memory is measured?

A. Copy ratio

B. Data ratio

C. Hit ratio

D. Cache ratio

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Hit ratio

Cache performance is explicitly measured by hit ratio, hits/(hits+misses), in PDF p.85. A hit means the requested word is found in cache (p.84), making this a direct measure of cache availability of requested data.

Why each choice is right or wrong

A - Incorrect: Copy ratio is not the performance quantity defined on p.85.

B - Incorrect: Data ratio is not the named hit/miss measure.

C - Correct: Hit ratio and its formula are explicitly provided (p.85).

D - Incorrect: “Cache ratio” is not the quantity’s stated name.

Primary note proof

Study-note section COA23

> Hit ratio=hits/accesses; miss ratio=1−hit ratio.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 85 | lines 2-8

> • The performance of cache memory is frequently measured in terms of a quantity called hit ratio
> Hit ratio = hit / (hit+miss)
> • The basic characteristic of cache memory is its fast access time
> • Therefore, very little or no time must be wasted when searching the words in the cache
> • The transformation of data from main memory to cache memory is referred to as a mapping process.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

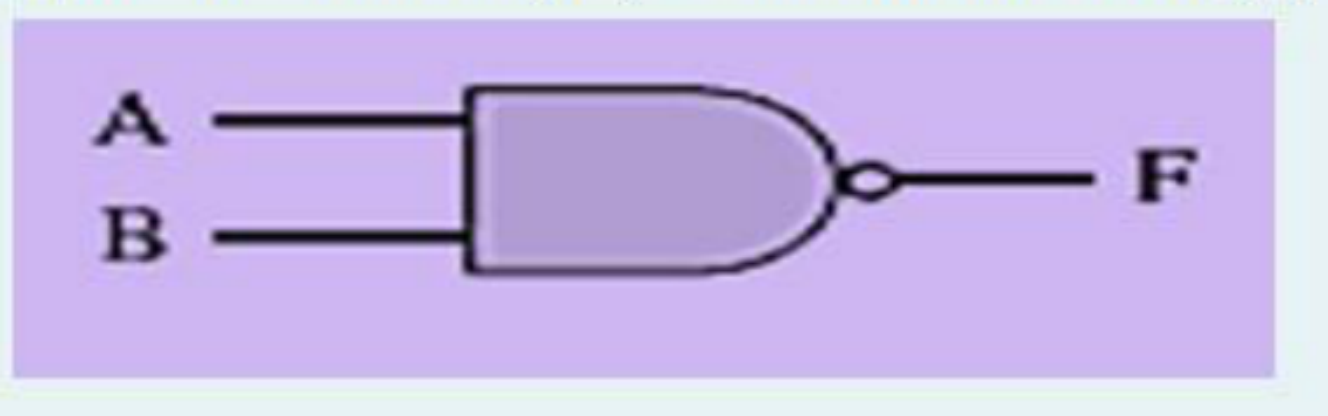
> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
> Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
> The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
> • The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
> • When CPU needs to access memory, the cache is examined
> • If the word is found in the cache, it is read from the fast memory
> • If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
> • When the CPU refers to memory and finds the word in cache, it is said to produce a hit
> • Otherwise, it is a miss

Question sources

Mock exam 1 · Question 9

Question 70 | EX2018Q100

Question number 100
What is the name of logic gate based on the following graphical symbol?



A. XOR

B. NOR

C. NAND

D. AND

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: NAND

The diagram shows an AND-shaped gate with a small output circle. This is a NAND gate; the circle indicates inversion, so F=NOT(A AND B).

Why each choice is right or wrong

A - Incorrect: XOR uses the OR-like body with an extra input curve, unlike the source symbol (p.7).

B - Incorrect: NOR uses an OR-shaped body with an output circle, not the flat-input AND body (p.7).

C - Correct: The AND body plus inversion circle is NAND (p.7).

D - Incorrect: AND lacks the output inversion circle visible in the question.

Primary note proof

Study-note section COA03

> A bubble at a gate output indicates inversion: an AND-shaped symbol with a bubble is NAND.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Question sources

Computer Science Exit Exam · 2018 · Question 100

Question 71 | SM0013

An interface that has multiple lines connecting the input/output module and peripheral device and assures multiple bits to be transferred at the same time is:

A. Bus interface

B. Parallel interface

C. One line interface

D. Serial interface

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Parallel interface

Parallel registers read/write bits simultaneously (PDF p.35); shift registers transfer serially (p.36), and a common bus supplies a line for each bit (p.40). Multiple lines carrying bits at the same time match a parallel interface by this distinction.

Why each choice is right or wrong

A - Incorrect: A bus describes a collection of lines; it does not specify serial versus parallel transfer as precisely as B (p.40).

B - Correct: Simultaneous multiple-bit transfer corresponds to the parallel operation defined on p.35.

C - Incorrect: One line cannot carry the described multiple separate bit signals simultaneously in this binary-line model.

D - Incorrect: Serial shifting transfers the stream sequentially rather than all bits simultaneously (p.36).

Primary note proof

Study-note section COA31

> Serial sends bits over time; parallel transfers multiple bits on multiple lines, without guaranteeing higher real-world speed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

> 3.9 Registers
> • A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
> • A register is a digital circuit used within the CPU to store one or more bits of data.
> • An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
> • In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
> • Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 36 | lines 2-9

> Shift Register
> • A shift register accepts and/or transfers information serially.
> • Data are input only to the leftmost flip-flop.
> • Shift registers can be used to interface to serial I/O devices.
> • With each clock pulse, data are shifted to the right, one position, and the rightmost bit is transferred out.
> • In addition, they can be used within the ALU to perform logical shift and rotate functions.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

> 4.2 Bus and Memory Transfers
> A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Question sources

MOE Computer Science Exit Exam · Question 4-1

National Computer Science Exit Exam · Question 96

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 228

Question 72 | SM0033

Which one of the following is a two-level cache having an internal and external caches?

A. Multilevel cache

B. Single level cache

C. Split cache

D. Unified cache

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer A: Multilevel cache.

A two-level hierarchy has more than one cache level, so multilevel is the matching category. COA26 distinguishes L1/L2 hierarchy levels from split/unified instruction-data organization and from internal/external placement. Describing one cache as internal and another as external does not make the pair split or unified; those terms answer a different classification question.

Why each choice is right or wrong

A - Correct: Two cache levels constitute a multilevel hierarchy, regardless of their internal or external placement.

B - Incorrect: Single-level cache contradicts the explicitly stated presence of two cache levels in the hierarchy.

C - Incorrect: Split describes separate instruction and data caches, not an internal-plus-external level arrangement.

D - Incorrect: Unified describes combined instruction/data caching, not how many internal or external levels exist.

Primary note proof

Study-note section COA26

> L1/L2 describe hierarchy levels; split cache means separate instruction/data caches and unified means combined.

Study-note section COA26

> Internal/external location is another classification, not synonymous with split/unified.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.

Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 79 | lines 3-13

6.1 Memory Hierarchy:
A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Question sources

MOE Computer Science Exit Exam · Question 10-2

National Computer Science Exit Exam · Question 26

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 229

Question 73 | SM0051

Which statement is NOT correct about computer architecture and organization?

A. An architecture can remain for a long time but its organization can change

B. Manufacturers offer computers with the same architecture but different organization

C. An architecture can be used for a long time encompassing different computer models

D. Different organizations cannot implement the same architecture

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Different organizations cannot implement the same architecture

Architecture describes programmer-visible behavior, while organization describes internal implementation. Several internal implementations can preserve the same architecture, allowing it to span models and remain stable while implementation details change. The false statement denies that possibility. The other choices describe compatible consequences of the distinction, without requiring a maximum architecture lifetime or a universal implementation-change rate.

Why each choice is right or wrong

A - Incorrect: Implementation can change while architecture stays stable.

B - Incorrect: Different organizations can implement one architecture.

C - Incorrect: One architecture can encompass different computer models over time.

D - Correct: This denies the supported possibility of different implementations of one architecture.

Primary note proof

Study-note section COA01

> Different organizations can implement the same architecture; neither definition imposes a maximum lifetime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 5 | lines 12-20

> Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
> Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

MOE Computer Science Exit Exam · Question 15-2

National Computer Science Exit Exam · Question 59

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 230

Question 74 | SM0059

_____ is a set of attributes that can be seen by a program and can have direct impact on the logical execution of a program.

A. Arrangement

B. Organization

C. Flow control

D. Architecture

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Architecture

Programmer-visible attributes that directly affect logical execution are the module's exact definition of computer architecture (PDF p.5). Organization instead comprises operational units and interconnections realizing those specifications.

Why each choice is right or wrong

A - Incorrect: Arrangement is not the defined programmer-visible attribute set.

B - Incorrect: Organization describes implementation units and interconnections (p.5).

C - Incorrect: Flow control is a function, not the definition of this attribute set.

D - Correct: Architecture exactly matches the definition and examples (p.5).

Primary note proof

Study-note section COA01

> Architecture describes programmer-visible behavior: instructions, registers, data types, address spaces and I/O conventions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 5 | lines 12-20

> Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
> Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

MOE Computer Science Exit Exam · Question 17-2

National Computer Science Exit Exam · Question 20

National Computer Science Exit Exam

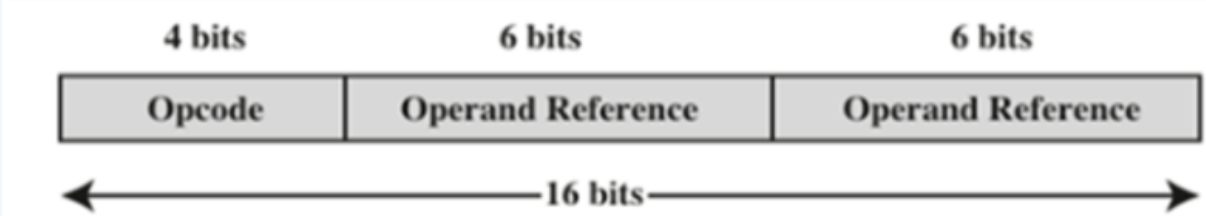
MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 231

Question 75 | SM0063

What is the purpose of an opcode that is as indicated in the figure below?
4 bits 6 bits 6 bits

[opcode] Operant etree [| Opsrand Rates|
<_ 16 bits -ccccxxcxor



A. It holds the destination address of an operand

B. It determines the operation to be done

C. It holds the source address of an operand

D. It holds the next instruction reference

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: It determines the operation to be done

The inspected diagram divides 16 bits into a 4-bit opcode and two 6-bit operand references. The module defines the operation part as specifying what operation is performed (PDF p.47); reference fields identify operands rather than changing that opcode role. Four opcode bits can distinguish up to 16 operation codes.

Why each choice is right or wrong

A - Incorrect: Destination reference belongs to an operand/address field, not the operation field (p.47).

B - Correct: Operation selection is the explicit opcode definition (p.47).

C - Incorrect: Source reference belongs to an operand/address field.

D - Incorrect: The module tracks the next instruction through PC (p.70), not this operation field.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Study-note section COA16

> A 4-bit opcode permits at most 16 encodings, not necessarily 16 implemented instructions.

Study-note section COA16

> A 6-bit operand-reference field has 64 encodings; what they designate depends on the machine.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

MOE Computer Science Exit Exam · Question 18-2

Question 76 | SM0091

Which one of the following registers holds the address of the next instruction to be fetched from memory?

A. Memory address register

B. Memory buffer register

C. Program counter

D. Instruction buffer register

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Program counter

PC holds the address of the instruction to execute next (PDF p.70). During fetch it is copied to AR and the instruction bits then enter IR while PC increments (p.55). Thus PC is the persistent next-instruction tracker.

Why each choice is right or wrong

A - Incorrect: MAR can temporarily receive the fetch address, but PC is the register that tracks the next instruction across cycles (p.55).

B - Incorrect: The module describes MDR as memory data storage (p.82), not the next-instruction address tracker.

C - Correct: Directly matches the PC definition and fetch transfers (pp.55,70).

D - Incorrect: An instruction buffer would hold instruction content; the module identifies IR for that role (p.48), not as the next-address tracker.

Primary note proof

Study-note section COA14

> PC identifies the next instruction to fetch.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be

executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1,
INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

6.2.1 Connection of RAM and CPU
Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Question sources

MOE Computer Science Exit Exam · Question 26-1

National Computer Science Exit Exam · Question 70

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 233

Question 77 | SM0201

Which one of the following is NOT an I/O command?

A. Status command

B. Output data command

C. Control command

D. Issue command

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: Issue command

The module enumerates four I/O command kinds: control, test, read and write (PDF pp.104-105). “Status command” is interpreted as the test/status-query kind, and “output data” as write. “Issue” is an action performed by CPU, not one of the four command kinds.

Why each choice is right or wrong

A - Incorrect: Status-query corresponds to the test command, which checks status conditions (p.104).

B - Incorrect: Output data corresponds to the write command (p.105).

C - Incorrect: Control command activates a peripheral and specifies its action (p.104).

D - Correct: Issue is not a command category in the four-type list (pp.104-105).

Primary note proof

Study-note section COA29

> Control commands start actions, test/status commands inspect readiness or errors, read commands bring device data toward the system, and write/output commands send system data to the device.

Study-note section COA29

> “Issue” describes sending a command, not a standard fourth command category.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 104 | lines 17-27

7.6 I/O commands
To execute an I/O related instruction, the CPU issues an address, specifying the particular I/O module and external device and an I/O command. Four types of I/O commands can be received by the I/O module when it is addressed by the CPU. They are
• A control command: is used to activate a peripheral and tell what to do. Example: a magnetic tape may be directed to rewind or move forward a record.
• A test command: is used to test various status conditions associated with an I/O module and its peripherals. The CPU wants to know the interested peripheral for use. It also wants to know the most recent I/O operation is completed and if any errors have occurred.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

• A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
• A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 103 | lines 2-9

Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
Address recognition:
just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Computer Organization and Architecture module | PDF page 101 | lines 10-16

7.4 Input/output Problems
Wide variety of peripherals and delivering different amounts of data per second Work at different speeds Send/receive data in different formats all slower than CPU and RAM. Hence I/O modules are used as a solution.
Input/output Module: It is the entity within a computer that is responsible for the control of one or more external devices and for the exchange of data between those devices and main memory and/or CPU.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 1-1

Computer Science Exit Exam · Sene 2016 · Question 31-3

Computer Science Exit Exam practice collection · Question 243

Question 78 | SM0230

The user who programs a computer in machine or assembly language must be aware of the following EXCEPT:

A. The register set

B. The type of data supported by the instructions

C. The analysis model

D. The memory structure

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: The analysis model

The machine-language programmer's architecture includes instruction formats, addressing modes, instruction sets and register organization (PDF p.57); visible data types and memory addressing are also named (p.5). An "analysis model" is not part of that architectural list.

Why each choice is right or wrong

A - Incorrect: Register organization is explicitly part of the programmer view (p.57).

B - Incorrect: Supported data representation is explicitly architectural (p.5).

C - Correct: Analysis model is absent from the programmer-facing attribute list; it is the unmatched option.

D - Incorrect: Memory addressing is part of the architecture (p.5), under the intended visible-memory interpretation.

Primary note proof

Study-note section COA02

> Machine-language programming depends on the register set, supported types and memory/addressing structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU

performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 5 | lines 12-20

Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 10-1

Computer Science Exit Exam practice collection · Question 244

Question 79 | SM0239

Which part of the computer system is responsible for converting machine-processed data to formats understandable by the user?

A. Communication components

B. Output unit

C. Storage unit

D. Input unit

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Output unit

Computational results must be recorded/displayed for users (PDF p.99); human-readable screen/display/printer devices provide that communication (p.100). This is the output role.

Why each choice is right or wrong

A - Incorrect: Communication devices exchange data with remote devices; the category is not specifically user-readable presentation (p.100).

B - Correct: Output devices record/display computed results in meaningful form for users (p.99).

C - Incorrect: Storage retains information; it is not the presentation role (p.91).

D - Incorrect: Input enters programs/data into the system, the reverse direction (p.99).

Primary note proof

Study-note section COA29

> Input units encode external information and output units present processed results.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 99 | lines 13-23

> The input/output subsystem of a computer, referred to as I/O, provides an efficient mode of communication between the central system and the outside environment. Programs and data must be entered into computer memory for processing and results obtained from computations must be recorded or displayed for users. A computer serves no useful purpose without the ability to receive information from an outside source and to transmit results in a meaningful form. I/O operations are accomplished through a wide assortment of external devices that provide a means of exchanging data between the external environment and the computer. An external device attaches to the computer by a link to an I/O module. The link is used to exchange control, status, and data between the I/O module and the external device. An external device connected to an I/O module also called Interface is often referred to as a peripheral device or simply a peripheral.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 91 | lines 8-15

> 6.4 Secondary memory
> Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
> • It stores information that is not necessarily in current use.
> • It is slower and having higher capacity than primary memory.
> • This kind of memory is large, slow and inexpensive.
> • It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
> • Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 13-1

Computer Science Exit Exam practice collection · Question 240

Question 80 | SM0242

In the usual memory hierarchy, which listed tier has the shortest access latency and greatest cost per stored bit?

A. Register

B. Magnetic disc

C. Main memory

D. Cache memory

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer A: Register

The usual speed and cost-per-bit hierarchy places registers above cache, main RAM and magnetic disk. Registers therefore have the shortest access latency and highest cost per bit among the choices. Faster tiers are generally smaller, so this statement does not compare the total purchase price of a complete device. Cache is fast, but remains below the register tier in this teaching hierarchy.

Why each choice is right or wrong

A - Correct: Registers are the fastest and most costly-per-bit tier among these choices.

B - Incorrect: Magnetic disk is a slower, high-capacity tier with lower cost per bit.

C - Incorrect: Main memory lies below registers and cache in the usual speed/cost hierarchy.

D - Incorrect: Cache is fast but below registers in the usual hierarchy.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA23

> Faster levels are generally smaller and more expensive per bit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of

memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 58 | lines 4-11

In the programming examples of the previous chapter, we have shown that memory locations are needed for storing pointers, counters, return addresses, temporary results and partial product during multiplication. Having to refer to memory locations for such applications is time consuming because memory access is the most time consuming operation in a computer. It is more convenient and more efficient to store these intermediate values in processor register. When a large number of registers are included in the CPU, It is most efficient to connect them through a common bus system. The registers communicate with each other not only for direct data transfers, but also while performing various micro operations.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 14-1

Computer Science Exit Exam practice collection · Question 241

Question 81 | SM0277

For a legacy internal PC disk connection using parallel ATA, which listed interface name matches that connection?

A. SCSI

B. iSCSI

C. IDE

D. USB

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: IDE

The note identifies legacy IDE/PATA as a storage connection, so IDE matches the specified parallel ATA context. SCSI is a different storage interface, iSCSI carries SCSI over IP, and USB commonly connects peripherals including external storage. Naming the legacy internal parallel interface resolves the missing era and connection-role constraints without asserting that one interface is universally standard for all PCs and all storage devices.

Why each choice is right or wrong

A - Incorrect: SCSI is a different storage-connection standard.

B - Incorrect: iSCSI transports SCSI commands over IP rather than using the specified local parallel interface.

C - Correct: IDE corresponds to the stated legacy PATA connection.

D - Incorrect: USB is a different peripheral interconnect, not parallel ATA.

Primary note proof

Study-note section COA32

> SATA is a storage interface; legacy IDE/PATA and SCSI represent other storage connections, while iSCSI carries SCSI commands over IP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 25-2

Computer Science Exit Exam practice collection · Question 245

Question 82 | SM0295

In the simple CPU-master memory-bus model, which bus carries the values read from and written to memory?

A. Control bus

B. Data bus

C. Uni-bus

D. Address bus

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Data bus

The data bus carries memory-transfer values. Reads move values from memory toward the CPU, and writes move values in the opposite direction. Address signals select a location and control signals coordinate timing and read/write operations. The revised stem asks about payload rather than bidirectionality in general, since aggregate control can also contain signals in both directions. Uni-bus names a bus arrangement rather than this payload-signal category.

Why each choice is right or wrong

A - Incorrect: Control signals coordinate the transfers rather than carrying their values.

B - Correct: The data bus carries read and write values.

C - Incorrect: This is not the address/data/control payload-signal category requested.

D - Incorrect: Address signals choose the memory location.

Primary note proof

Study-note section COA12

Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Study-note section COA12

In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

• A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
• A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 102 | lines 20-24

I/O module must have the capability to engage in communication with the CPU and external device. Thus CPU communication involves
Command decoding: The I/O module accepts commands from the CPU carried on the control bus.
Data: data are exchanged between the CPU and the I/O module over data bus

Computer Organization and Architecture module | PDF page 103 | lines 2-9

Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
Address recognition:
just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Question sources

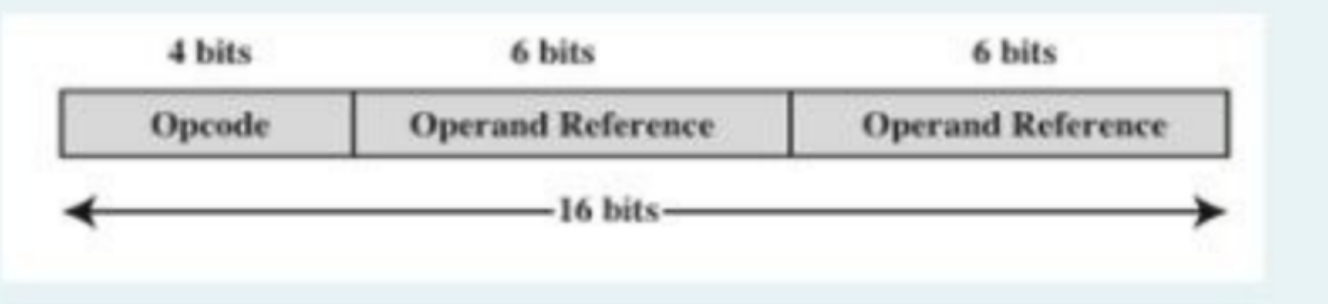
Computer Science Exit Exam · Sene 2016 · Question 31-2

Computer Science Exit Exam practice collection · Question 242

Question 83 | SS077

What is the purpose of an opcode that is as indicated in the figure below?
4 bits 6 bits 6 bits

<j o bits --_____»



A. It determines the operation to be done

B. It holds the destination address of an operand

C. It holds the next instruction reference

D. It holds the source address of an operand

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: It determines the operation to be done

The instruction has a 4-bit Opcode followed by two 6-bit Operand Reference fields. Opcode selects the operation.

Why each choice is right or wrong

A - Correct: Matches the operation-code definition (p.47).

B - Incorrect: A destination reference is an operand field, not the opcode.

C - Incorrect: Next-instruction location is tracked by PC (p.70), not this operation field.

D - Incorrect: Source address belongs to an operand reference, not the operation code (p.47).

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The

most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

National Computer Science Exit Exam · Question 80

National Computer Science Exit Exam

Question 84 | AM0704

Which of the following is a common way of communicating with I/O devices?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. All of the above

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: All of the above

Memory-mapped I/O shares memory and device address space (PDF p.105); DMA transfers blocks directly (p.107); interrupt I/O requests CPU service when ready (p.106). They are all supported I/O mechanisms, though memory mapping is an addressing choice and the others are transfer/control techniques.

Why each choice is right or wrong

A - Incorrect: Memory-mapped I/O is explicitly described (p.105).

B - Incorrect: DMA is explicitly described (p.107).

C - Incorrect: Interrupt-driven I/O is explicitly described (p.106).

D - Correct: All three are legitimate mechanisms under the broad “communicating” wording.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA31

> Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

> 7.8 I/O Mapping
> When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
> • Memory mapped I/O
> • Devices and memory share an address space
> • I/O looks just like memory read/write
> • No special commands for I/O
> • Large selection of memory access commands available
> • Isolated I/O
> • Separate address spaces
> • Need I/O or memory select lines
> • Special commands for I/O and Limited set

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer’s motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Question sources

Computer Science MCQ practice collection · Question 704

Question 85 | AM0705

In memory-mapped I/O, I/O devices are mapped into the:

A. Main memory address space

B. Cache memory address space

C. Virtual memory address space

D. Secondary memory address space

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: Main memory address space

Memory-mapped I/O makes devices and memory share an address space, with I/O treated like memory reads/writes (PDF p.105). The relevant address space is the memory address space, not a separate cache or secondary-storage space.

Why each choice is right or wrong

A - Correct: Devices share the memory address space (p.105).

B - Incorrect: The module does not map device registers into a separate cache address space.

C - Incorrect: The mapping definition does not depend on a process virtual address space.

D - Incorrect: Secondary-storage addresses are not the shared CPU memory addresses described on p.105.

Primary note proof

Study-note section COA31

Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Study-note section COA31

Isolated/port-mapped I/O has a separate port space and special instructions such as IN/OUT.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Question sources

Computer Science MCQ practice collection · Question 705

Question 86 | AM0706

Which technique allows data to be transferred between an I/O device and memory without involving the CPU?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. I/O polling

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Direct memory access (DMA)

DMA transfers each word directly to/from memory without passing through CPU (PDF p.108). “Without involving” must mean during the actual word transfers, since CPU setup and completion involvement are explicit on the same page.

Why each choice is right or wrong

A - Incorrect: Memory mapping specifies how I/O is addressed, not a CPU-bypass data engine (p.105).

B - Correct: DMA directly transfers the block with CPU only at beginning/end (p.108).

C - Incorrect: The CPU still executes the interrupt-driven data transfer (p.106).

D - Incorrect: Polling keeps CPU checking device status and doing transfers (p.104).

Primary note proof

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> DMA does not eliminate CPU involvement in configuration, ownership, address mapping or completion.

Study-note section COA31

> Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer’s motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

> When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
> • Whether to read or write
> • Device address
> • Starting address of memory block for data
> • Amount of data to be transferred The CPU carries on with other work.
> • Thus DMA controller steals the CPU‟s work of I/O operation.
> • The DMA module transfers the entire block of data
> • One word at a time, directly to or from memory, without going through CPU.
> When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Question sources

Computer Science MCQ practice collection · Question 706

Question 87 | AM0707

Which method is used to handle I/O operations that require immediate attention from the CPU?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. I/O polling

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Interrupt-driven I/O

In interrupt-driven I/O the device requests CPU service when ready; CPU can work on other instructions until then (PDF p.106). The priority-interrupt discussion on that page services the higher-priority simultaneous request first. This matches event-driven attention, without guaranteeing zero response latency.

Why each choice is right or wrong

A - Incorrect: Memory mapping selects addresses and does not itself notify the CPU (p.105).

B - Incorrect: DMA offloads data movement (p.107); it is not the general CPU-service notification technique.

C - Correct: The device interrupts CPU requesting service when ready (p.106).

D - Incorrect: Polling requires the CPU to discover readiness through repeated checks (p.104).

Primary note proof

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA31

> Interrupt priority selects among simultaneous requests, using schemes such as polling, daisy chains or priority hardware.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of

actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Question sources

Computer Science MCQ practice collection · Question 707

Question 88 | AM0708

Which I/O technique involves the CPU continuously checking the status of an I/O device for data transfer completion?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. I/O polling

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: I/O polling

Programmed I/O explicitly makes CPU periodically check I/O status until completion, because the module does not interrupt CPU (PDF p.104). This is the polling behavior described by D.

Why each choice is right or wrong

A - Incorrect: Memory mapping is the address arrangement and does not require repeated status checks (p.105).

B - Incorrect: DMA permits CPU other work while moving a block (p.108).

C - Incorrect: Interrupt I/O notifies CPU when ready instead of requiring periodic checking (p.106).

D - Correct: CPU checks status bits periodically until completion (p.104).

Primary note proof

Study-note section COA30

Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Question sources

Computer Science MCQ practice collection · Question 708

Question 89 | AM0709

Which standard I/O interface is commonly used in UNIX-like operating systems?

A. ASCII

B. USB

C. RS-232

D. POSIX

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: POSIX.

COA32 identifies POSIX as a software interface standard with open/read/write/close operations. That is the software I/O interface sense appropriate to UNIX-like operating systems. ASCII is an encoding, while USB and RS-232 describe hardware interconnect/interface roles. Their availability on a UNIX-like machine does not make them the standard OS software interface requested by the alternatives.

Why each choice is right or wrong

A - Incorrect: ASCII defines character encoding, not the UNIX-like software interface for opening, reading and writing resources.

B - Incorrect: USB is a peripheral interconnect, not the software API standard identified by this OS-level wording.

C - Incorrect: RS-232 defines a serial electrical/interface standard rather than the UNIX-like software I/O API.

D - Correct: POSIX specifies software interface operations including open, read, write and close for this context.

Primary note proof

Study-note section COA32

POSIX is a software interface standard with operations such as open/read/write/close; ASCII is a character encoding.

Study-note section COA32

USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 112 | lines 3-19

A data communication processor is an I/O processor that distribute and collect data from many remote terminals connected through telephone and other communication lines it is a specialized I/O processor designed to communicate directly with data communication network. The most striking difference between I/O processor and data communication processor is in the way processor communicate with the I/O devices.an I/O processor communicate with the peripherals through common I/O bus that is comprised of many data and control lines. All peripherals share common bus and use it to transfer information to and from I/O processor. A data communication processor communicates with each terminal through a single pairs of wires.
Data can be transmitted between two points in three different modes: simplex, half duplex

and full duplex.
A simplex line carries information in one direction only. This mode is seldom used in data communication because the receiver cannot communicate to the transmitter with indicate the occurrence of error examples of simplex transmission are radio and television broadcasting. Half duplex transmission system is one that have capable of transmitting in both direction but, data can transmitted in only one direction at a time. A pair of wire is needed for mode. A full duplex transmission can send and receive data in both directions simultaneously. This

Question sources

Computer Science MCQ practice collection · Question 709

Question 90 | AM0710

Which listed interface is a traditional point-to-point serial electrical/interface standard?

A. USB

B. Ethernet

C. RS-232

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: RS-232

RS-232 specifies a traditional point-to-point serial electrical/interface standard. USB also uses serial communication, so serial alone cannot distinguish them. The revised requirement identifies the traditional point-to-point interface category rather than treating every serial link as equivalent. Ethernet connects network devices and HDMI carries digital audio and video. Their connection roles differ from the interface class requested.

Why each choice is right or wrong

A - Incorrect: USB is a serial peripheral interconnect, but not the traditional point-to-point electrical standard requested.

B - Incorrect: Ethernet provides networking connectivity.

C - Correct: RS-232 matches the traditional point-to-point serial interface category.

D - Incorrect: HDMI carries digital audio and video.

Primary note proof

Study-note section COA32

USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

HDMI carries digital video and audio.

Study-note section COA32

Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 101 | lines 2-9

Asynchronous data transfer
The internal operation in digital system is synchronized by means of clock pulse supplied by a common pulse generator. Two units such as cpu and an I/O are designed independently of each other, if the register in the interface share a common clock with the cpu registers, the transfer between the two units is called to be synchronous.in most case the internal timing in each unit is independent from the other in that each uses its own private clock for internal register.in such cases the two units are said to be asynchronous to each other. This approach is widely used in most computers systems.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 36 | lines 2-9

Shift Register
• A shift register accepts and/or transfers information serially.
• Data are input only to the leftmost flip-flop.
• Shift registers can be used to interface to serial I/O devices.
• With each clock pulse, data are shifted to the right, one position, and the rightmost bit is transferred out.
• In addition, they can be used within the ALU to perform logical shift and rotate functions.

Computer Organization and Architecture module | PDF page 112 | lines 3-19

A data communication processor is an I/O processor that distribute and collect data from many remote terminals connected through telephone and other communication lines it is a specialized I/O processor designed to communicate directly with data communication network. The most striking difference between I/O processor and data communication processor is in the way processor communicate with the I/O devices.an I/O processor

communicate with the peripherals through common I/O bus that is comprised of many data and control lines. All peripherals share common bus and use it to transfer information to and from I/O processor. A data communication processor communicates with each terminal through a single pairs of wires.
Data can be transmitted between two points in three different modes: simplex, half duplex and full duplex.
A simplex line carries information in one direction only. This mode is seldom used in data communication because the receiver cannot communicate to the transmitter with indicate the occurrence of error examples of simplex transmission are radio and television broadcasting.
Half duplex transmission system is one that have capable of transmitting in both direction but, data can transmitted in only one direction at a time. A pair of wire is needed for mode.
A full duplex transmission can send and receive data in both directions simultaneously. This

Question sources

Computer Science MCQ practice collection · Question 710

Question 91 | AM0711

Which standard I/O interface is widely used for connecting external storage devices, such as hard drives and flash drives?

A. SATA

B. USB

C. Ethernet

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: USB.

COA32 explicitly describes USB as a peripheral interconnect used by external storage. This directly matches the stem's external hard-drive and flash-drive examples. SATA is described more generally as a storage interface, Ethernet connects network devices, and HDMI carries video/audio. The external peripheral examples therefore make USB the supported conventional choice among these options.

Why each choice is right or wrong

A - Incorrect: SATA is a storage interface, but USB is the note's explicit match for the external-peripheral examples.

B - Correct: The note explicitly lists external storage among USB peripheral connections, matching the supplied examples.

C - Incorrect: Ethernet connects network devices rather than being the named external hard-drive/flash-drive peripheral interconnect.

D - Incorrect: HDMI carries digital video and audio, rather than providing the external-storage interface requested.

Primary note proof

Study-note section COA32

> USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

> HDMI carries digital video and audio.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

> 6.4.1 Magnetic Disk
> • Magnetic disks are the foundation of external memory on virtually all computer systems.
> • A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
> • More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
> • It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
> • We can randomly access the data.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science MCQ practice collection · Question 711

Question 92 | AM0712

Which listed interface is specifically a storage-drive interface, rather than a general peripheral interconnect, networking interface or audiovisual link?

A. SATA

B. USB

C. Ethernet

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: SATA

SATA is specifically a storage interface. USB supplies general peripheral connectivity, Ethernet connects network devices and HDMI carries digital audio/video. The connection role distinguishes these alternatives; a vague high-speed description would not, because several interfaces can be fast in suitable versions. The question therefore asks for a storage-specific interface category without assuming that speed alone uniquely identifies it.

Why each choice is right or wrong

A - Correct: SATA is the storage-specific interface in the list.

B - Incorrect: USB provides a different peripheral-interconnect role.

C - Incorrect: Ethernet serves networking rather than this drive connection.

D - Incorrect: HDMI carries audiovisual data rather than this storage link.

Primary note proof

Study-note section COA32

> SATA is a storage interface; legacy IDE/PATA and SCSI represent other storage connections, while iSCSI carries SCSI commands over IP.

Study-note section COA32

> USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

> HDMI carries digital video and audio.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

> 4.2 Bus and Memory Transfers
> A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

> 6.4.1 Magnetic Disk
> • Magnetic disks are the foundation of external memory on virtually all computer systems.
> • A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
> • More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
> • It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
> • We can randomly access the data.

Question sources

Computer Science MCQ practice collection · Question 712

Question 93 | AM0713

Which standard I/O interface is primarily used for audio and video transmission between devices?

A. SATA

B. USB

C. Ethernet

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: HDMI.

COA32 explicitly identifies HDMI as carrying digital video and audio. That directly matches the primary transmission purpose asked here. SATA serves storage, USB serves general peripherals, and Ethernet serves network connections. Other links can transport encoded media in broader systems, but the stem asks the primary interface role, for which HDMI is the specific supported choice.

Why each choice is right or wrong

A - Incorrect: SATA is identified as a storage interface rather than the primary audiovisual connection in this list.

B - Incorrect: USB is a general peripheral interconnect; the note specifically assigns the audiovisual role to HDMI.

C - Incorrect: Ethernet is a network connection rather than the specifically identified digital audiovisual interface.

D - Correct: HDMI is explicitly described as carrying digital video and audio between devices.

Primary note proof

Study-note section COA32

> HDMI carries digital video and audio.

Study-note section COA32

> USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science MCQ practice collection · Question 713

Question 94 | AM0714

Which technique divides instruction processing into stages so that different stages work on successive instructions during the same cycle?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer A: Pipelining

Pipelining overlaps stages of successive instructions: one may be fetched while another is decoded or executed. Superscalar execution instead concerns issuing multiple instructions to functional units in one cycle; it can coexist with a pipeline but is not the definition of the staged organization. Caching supplies nearby copies of memory blocks, and branch prediction guesses future control flow.

Why each choice is right or wrong

A - Correct: Pipelining divides processing into stages and overlaps those stages across successive instructions.

B - Incorrect: Caching retains memory-block copies; it is not the staged instruction organization described.

C - Incorrect: Superscalar architecture provides multiple-issue execution resources rather than defining stage overlap itself.

D - Incorrect: Branch prediction guesses direction or targets, rather than dividing instruction processing into pipeline stages.

Primary note proof

Study-note section COA35

> A pipeline overlaps stages of different instructions; it primarily improves throughput.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science MCQ practice collection · Question 714

Question 95 | AM0715

Which technique improves the performance of a processor by storing frequently accessed data closer to the processor?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Caching

Caching keeps copies of memory blocks near the CPU, exploiting temporal and spatial locality to reduce many memory-access delays. Pipelining overlaps instruction stages, superscalar execution provides multiple-issue resources, and branch prediction guesses control flow. These mechanisms may coexist, but the stated nearby data-copy storage role specifically identifies caching.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps instruction stages rather than creating a data buffer (p.116).

B - Correct: Cache is explicitly the fast intermediate RAM buffer that supplies hits (p.84).

C - Incorrect: Superscalar execution issues multiple instructions to functional units; it is not a memory-copy storage tier.

D - Incorrect: Branch prediction guesses control-flow direction or targets; it does not name a nearby memory-block buffer.

Primary note proof

Study-note section COA23

> Cache copies memory blocks near the CPU.

Study-note section COA23

> Temporal locality reuses recent data; spatial locality uses nearby locations.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Study-note section COA36

Branch prediction guesses direction/target and continues speculatively; wrong guesses discard work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science MCQ practice collection · Question 715

Question 96 | AM0716

Which architecture allows multiple instructions to be executed in parallel by having multiple execution units?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Superscalar architecture.

COA36 defines superscalar execution as issuing multiple instructions to multiple functional units in a cycle when dependencies permit. This matches the multiple execution units in the stem. COA35 distinguishes that multiple-issue capability from ordinary pipelining, which overlaps stages. Caching supplies reused memory data and branch prediction speculates about control flow; neither names the multiple-issue architecture.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages but does not by itself provide multiple-instruction issue to several execution units.

B - Incorrect: Caching retains nearby copies of memory blocks rather than defining parallel instruction issue.

C - Correct: Superscalar execution uses multiple functional units to issue several independent instructions in a cycle.

D - Incorrect: Branch prediction guesses future control flow, rather than naming the architecture with multiple execution units.

Primary note proof

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Study-note section COA35

> A pipeline overlaps stages of different instructions; it primarily improves throughput.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 716

Question 97 | AM0717

Which technique predicts the outcome of conditional branches to improve instruction execution in a processor?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Branch prediction.

COA36 says branch prediction guesses a branch's direction/target and continues speculatively, discarding work when the guess is wrong. Predicting the outcome of a conditional branch is exactly that role. Pipelining overlaps processing stages, caching retains reusable data, and superscalar execution issues several instructions; those mechanisms can coexist with prediction but do not replace its definition.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps instruction stages; it does not itself predict the outcome of a conditional branch.

B - Incorrect: Caching retains reusable memory contents, rather than selecting the predicted direction of conditional control flow.

C - Incorrect: Superscalar execution provides multiple issue; it does not name the conditional-outcome prediction technique.

D - Correct: Branch prediction guesses branch direction or target so speculative execution can continue before resolution.

Primary note proof

Study-note section COA36

> Branch prediction guesses direction/target and continues speculatively; wrong guesses discard work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

> 8.3 Pipelining
> Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The

decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 717

Question 98 | AM0718

Which technique improves memory access performance by organizing memory into hierarchical levels?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Virtual memory

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Caching

Cache is the fast memory-block-copy tier used to reduce the cost of repeated or nearby CPU accesses. Virtual memory instead supplies translated address spaces and can use backing storage for nonresident pages. Pipelining and superscalar execution organize instruction processing, so neither names the memory-hierarchy buffer requested. The answer follows the storage role rather than a general claim that only one mechanism can improve performance.

Why each choice is right or wrong

A - Incorrect: Pipelining divides computational tasks into stages (p.116), not memory storage levels.

B - Correct: Cache supplies the fast tier in the hierarchy and avoids main-memory access on hits (pp.79,84).

C - Incorrect: Superscalar execution concerns multiple instruction issue, not a memory-hierarchy storage tier.

D - Incorrect: Virtual memory supplies an apparent larger memory space (p.90), not the fast-cache-hit mechanism.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA23

> Cache copies memory blocks near the CPU.

Study-note section COA27

> Virtual memory gives processes translated address spaces and protection; backing storage can hold nonresident pages.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
> Figure 6.1 memory hierarchy
> • Design constraints: How much? How fast? How expensive?
> • Faster access time, greater cost per bit
> • Greater capacity, smaller cost per bit,
> • Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main

memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 90 | lines 9-23

Concept of Virtual Memory
A programmer can write a program which requires more memory space than the capacity of the main memory. Such a program is executed by virtual memory technique.
The program is stored in the secondary memory. The memory management unit (MMU) transfers the currently needed part of the program from the secondary memory to the main memory for execution.
This to and fro movement of instructions and data (parts of a program) between the main memory and the secondary memory is called Swapping.
Address Space And Memory Space.
Virtual address is the address used by the programmer and the set of such addresses is called the address space or virtual memory.
An address in main memory is called a location or physical address. The set of such locations in main memory is called the memory space or physical memory.
CPU generated logical address consisting of a logical page number plus the location within that page (x).

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Question sources

Computer Science MCQ practice collection · Question 718

Question 99 | AM0719

Which processor technique issues multiple independent instructions to multiple functional units in one clock cycle, when dependencies permit?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Instruction-level parallelism

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Superscalar architecture

Superscalar execution issues multiple instructions in a cycle when the machine has suitable resources and the instructions' dependencies permit it. Pipelining overlaps stages of different instructions but need not issue several at once. Caching addresses memory access, while instruction-level parallelism is the broader opportunity exploited by several mechanisms. The revised stem asks about multiple issue rather than incorrectly claiming that wider instruction supply necessarily reduces individual memory latency.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages but does not by itself require multiple issue.

B - Incorrect: Caching retains memory-block copies and is not the multiple-issue technique.

C - Correct: Superscalar execution issues multiple instructions to multiple units in a cycle.

D - Incorrect: Instruction-level parallelism is the broader property exploited rather than this specific issue mechanism.

Primary note proof

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Study-note section COA35

> "Fetch/decode one instruction while executing another" describes pipelining, whereas issuing several instructions in one cycle requires multiple-issue resources.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

> 8.3 Pipelining
> Pipelining is an implementation technique where multiple instructions are overlapped in

execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Question sources

Computer Science MCQ practice collection · Question 719

Question 100 | AM0720

Which technique translates source or intermediate code into target code during program execution, before the translated portion runs?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Just-in-time (JIT) compilation

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Just-in-time (JIT) compilation

JIT compilation translates code while the program is running, often compiling an intermediate representation before executing that portion. This timing distinguishes it from ahead-of-time translation. Pipelining overlaps processing stages, caching retains copies of data or instructions, and superscalar execution issues multiple instructions to execution resources. Those mechanisms do not name runtime source-to-target compilation.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages of executing instructions; it does not translate high-level or intermediate code.

B - Incorrect: Caching retains data or instruction copies; it does not perform the compilation described in the stem.

C - Incorrect: Superscalar architecture issues multiple instructions; it does not translate a source language into target code.

D - Correct: JIT is the only offered compiler; it translates during program execution before the compiled portion executes.

Primary note proof

Study-note section COA02

> JIT compilation happens during program execution, often translating intermediate code.

Study-note section COA02

> Translation before execution alone does not distinguish JIT from an ahead-of-time compiler.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 720

Question 101 | AM0721

Which technique improves performance by predicting the next set of instructions to be executed based on past execution patterns?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Branch prediction.

Predicting which instruction stream follows a branch is the direction/target prediction described in COA36. Using past patterns is consistent with that prediction role, whereas pipeline overlap, cached memory copies and multiple instruction issue address different mechanisms. Prediction remains speculative: a wrong guess discards work rather than changing the program's required architectural result.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages and requires instruction supply, but it is not itself next-path prediction.

B - Incorrect: Caching retains copies for reuse; a cached instruction is not a prediction of future control flow.

C - Incorrect: Superscalar issue increases concurrent instruction execution rather than identifying the upcoming branch path.

D - Correct: Predicting branch direction and targets identifies the likely future instruction stream for speculative execution.

Primary note proof

Study-note section COA36

> Branch prediction guesses direction/target and continues speculatively; wrong guesses discard work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the

instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 721

Question 102 | AM0722

Which technique improves performance by dynamically reordering instructions to maximize instruction-level parallelism?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Out-of-order execution

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Out-of-order execution.

COA36 describes out-of-order execution as scheduling ready operations before stalled older ones while preserving architectural results and exceptions. This is dynamic reordering to exploit instruction-level parallelism. Superscalar issue can provide execution width but does not itself define such reordering. Pipelining overlaps stages and caching improves reuse; neither names the readiness-based dynamic scheduling mechanism.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages; dynamic instruction reordering requires an additional readiness-based execution mechanism.

B - Incorrect: Caching keeps reusable memory copies and does not itself reorder instruction execution.

C - Incorrect: Superscalar issue permits multiple instructions per cycle but does not necessarily require out-of-order scheduling.

D - Correct: Out-of-order execution schedules ready instructions around stalled older ones while preserving required results.

Primary note proof

Study-note section COA36

> Out-of-order execution schedules ready operations before stalled older ones while preserving architectural results and exception behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

> The following are important points about parallel processing:
> • A parallel processing system is able to perform concurrent data processing to achieve faster execution time.

• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 722

Question 103 | AM0723

Which technique improves performance by dynamically allocating processor resources based on program behavior?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Dynamic resource allocation

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Dynamic resource allocation.

COA36 explicitly defines dynamic resource allocation as assigning available internal entries/units at runtime. That definition directly fits the stem's allocation based on behavior during execution. Pipelining describes overlapping stages, caching describes memory reuse, and superscalar execution describes issuing several instructions. They may use allocated resources, but none is the specific allocation technique named by the question.

Why each choice is right or wrong

A - Incorrect: Pipelining defines overlapping instruction stages rather than the general assignment of runtime internal resources.

B - Incorrect: Caching retains memory copies; it does not name the runtime assignment of processor entries and units.

C - Incorrect: Superscalar execution names multiple issue, rather than the allocation process used to assign available resources.

D - Correct: The note explicitly defines this as assigning available internal processor entries and units at runtime.

Primary note proof

Study-note section COA36

> Dynamic resource allocation assigns available internal entries/units at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

> The following are important points about parallel processing:
> • A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
> • The system may have two or more ALUs and be able to execute two or more instructions at the same time

• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 59 | lines 2-15

The control unit that operates the CPU bus system directs the information flow through the register and ALU by selecting the various components in the system. For example, to perform the operation
R1 R2 + R3, the control must provide binary selection variables to the following selector inputs:
1. MUX A selector (SELA): to place the content of R2 into bus A.
2. MUX selector (SELB): to place the content of R3 into bus B.
3. ALU operation selector (OPR): to provide the arithmetic add A +B.
4. Decoder destination selector (SELD): to transfer the content of output bus in to R1.
The four control selection variables are generated in the control unit and must be available at the beginning of a clock cycle. The data from the two source registers propagates through the gates in the multiplexers and the ALU, to the output bus, and into the inputs of the destination register, all during the clock cycle interval. Then, when the next clock transition occurs, the binary information from the output bus is transferred into R1. To achieve a fast response

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 723

Question 104 | AM0724

Which listed item is a fundamental logic building block from which computer hardware is constructed?

A. CPU

B. Memory

C. Hard disk

D. Logic gate

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Logic gate

Logic gates perform elementary Boolean functions and are fundamental building blocks of digital hardware. A CPU, memory system and hard disk are larger components built from many lower-level mechanisms. The final option identifies the logic-level building block. The question explicitly states the level of abstraction so that basic does not mean an arbitrary major system component.

Why each choice is right or wrong

A - Incorrect: A CPU is a major assembled component rather than a fundamental logic building block.

B - Incorrect: Memory is a storage component rather than the requested logic-level building block.

C - Incorrect: A hard disk is a complex storage device.

D - Correct: A logic gate is the fundamental logic-level building block requested.

Primary note proof

Study-note section COA01

> Gates are fundamental logic building blocks; distinguish that level from CPU, memory, disk and motherboard components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal.
> The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an

output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Computer Science MCQ practice collection · Question 724

Question 105 | AM0725

Which component is responsible for executing instructions and performing calculations in a computer?

A. CPU

B. Memory

C. Hard disk

D. Motherboard

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: CPU

The CPU contains the control unit, ALU and execution registers. Control sequences operations and transfers, while the ALU performs arithmetic and logic, supporting the CPU's instruction-processing role. Main memory supplies addressed working storage, a hard disk supplies persistent storage, and the motherboard connects components. Those roles distinguish the processing component from the storage and interconnection choices.

Why each choice is right or wrong

A - Correct: CPU's processing/execution function is explicit (pp.6,57).

B - Incorrect: Main memory stores data (p.6), rather than being the arithmetic execution unit.

C - Incorrect: Disk is secondary storage (p.94), not the processing unit.

D - Incorrect: The motherboard physically interconnects components; the CPU carries out the instruction execution and calculations.

Primary note proof

Study-note section COA01

> The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Study-note section COA01

> A motherboard physically connects components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers

and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science MCQ practice collection · Question 725

Question 106 | AM0726

Which component retains stored data and instructions without power? Here Memory means ordinary volatile main RAM, and CPU excludes separate firmware storage.

A. CPU

B. Memory

C. Hard disk

D. Motherboard

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Hard disk

An HDD preserves magnetic patterns without continuous power, so the hard disk retains the stored data and instructions. The question now restricts Memory to ordinary volatile main RAM, avoiding the broader meaning that would include ROM or flash. CPU and motherboard name processing and interconnection components rather than the persistent storage mechanism sought. These scope conditions make the intended retention comparison definite.

Why each choice is right or wrong

A - Incorrect: The CPU is not the specified persistent storage component.

B - Incorrect: Ordinary main RAM is volatile under the stated meaning.

C - Correct: An HDD retains its stored bits without continuous power.

D - Incorrect: The motherboard interconnects components rather than naming the persistent storage mechanism.

Primary note proof

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Study-note section COA24

> Both are ordinarily volatile.

Study-note section COA05

> Ordinary flip-flops and SRAM lose state when power is removed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

> 6.4 Secondary memory
> Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
> • It stores information that is not necessarily in current use.
> • It is slower and having higher capacity than primary memory.
> • This kind of memory is large, slow and inexpensive.
> • It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
> • Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

> 6.2 Main Memory
> The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

> DYNAMIC RAM (DRAM)
> is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
> STATIC RAM (SRAM)
> In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major

parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 81 | lines 6-16

Static RAM:
• a bit of data is stored using the state of a flip-flop.
• Applying power is enough (no need for refreshing).
• Retains value indefinitely, as long as it is kept powered.
• Mostly uses to create cache memory of CPU
• Faster and more expensive than DRAM.
ROM
ROM is used for storing programs that are permanently resident in the computer and for tables of constants that do not change in value once the production of the computer is completed The ROM portion of main memory is needed for storing an initial program called bootstrap loader, which is to start the computer software operating when power is

Question sources

Computer Science MCQ practice collection · Question 726

Question 107 | AM0727

The motherboard of a computer is responsible for:

A. Executing instructions

B. Storing data

C. Connecting different components

D. Performing calculations

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Connecting different components.

COA01 explicitly states that a motherboard physically connects components. The same section locates the ALU, control unit and registers in the CPU; computation and instruction execution therefore belong to processing components rather than the board's defining role. Memory provides data storage. The question asks the motherboard's responsibility, for which physical connection is the specific supported answer.

Why each choice is right or wrong

A - Incorrect: Instructions are executed by the CPU; that is distinct from the motherboard's physical connection role.

B - Incorrect: Data storage belongs to memory/storage components, rather than defining what the motherboard itself does.

C - Correct: The note explicitly identifies the motherboard as physically connecting the computer's components.

D - Incorrect: Calculations belong to CPU arithmetic/logic hardware rather than the motherboard's defining connection function.

Primary note proof

Study-note section COA01

> A motherboard physically connects components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes

the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science MCQ practice collection · Question 727

Question 108 | AM0728

Which component supplies the addressable main working store for a running program, distinct from individual CPU execution registers and persistent disk storage?

A. CPU

B. Main memory (RAM)

C. Hard disk

D. Motherboard

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Main memory (RAM)

Main RAM holds the addressed working instructions and data used by a running program. CPU registers hold particular fetched instructions, operands or results for immediate execution, and disks retain information persistently. The motherboard connects these components. Specifying the main working-store role prevents the broad temporary-storage description from also fitting CPU registers or cache.

Why each choice is right or wrong

A - Incorrect: The CPU executes operations and has immediate registers, but is not the main addressed RAM store requested.

B - Correct: Main RAM is the addressable working store for the running program’s instructions and data.

C - Incorrect: A hard disk supplies persistent storage rather than the main RAM working store.

D - Incorrect: The motherboard provides physical component connections rather than this storage role.

Primary note proof

Study-note section COA02

In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA24

DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA14

Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA28

It does not need continuous power to preserve stored bits.

Study-note section COA01

A motherboard physically connects components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

6.2 Main Memory
The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)

• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science MCQ practice collection · Question 728

Question 109 | AM0729

Which type of software controls and manages the hardware resources of a computer?

A. Operating system

B. Application software

C. Programming language

D. Compiler

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Operating system.

COA02 directly defines the operating system as managing resources and providing services. Applications solve user tasks, and compilers translate source into target code. A programming language is a means of expressing programs rather than the software resource manager. Therefore the operating system is the single option whose stated role matches controlling and managing computer hardware resources.

Why each choice is right or wrong

A - Correct: The operating system is explicitly defined as the resource manager and provider of system services.

B - Incorrect: Applications solve user tasks using system services rather than serving as the general hardware resource manager.

C - Incorrect: A programming language expresses programs; it is not itself the system software managing hardware resources.

D - Incorrect: A compiler translates source code into target code rather than performing general OS resource management.

Primary note proof

Study-note section COA02

An operating system manages resources and provides services.

Study-note section COA02

Applications solve user tasks.

Study-note section COA02

A compiler translates source into target code; assembly is a symbolic form of machine instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 2-7

It must be mapped onto an actual (physical) main memory address by the operating system using mapper.
If the page is present in the main memory, CPU gets the required data from the main memory.
If the mapper detects that the requested page is not present in main memory, a page fault occurs and the page must be read from secondary storage into a page frame in main memory.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline

• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Computer Science MCQ practice collection · Question 729

Question 110 | AM0730

Which software allows users to perform specific tasks or functions on a computer?

A. Operating system

B. Application software

C. Programming language

D. Compiler

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: Application software.

COA02 states that applications solve user tasks. This directly matches software allowing users to perform particular tasks or functions. The operating system instead manages resources and provides services; a compiler translates programs, and a programming language expresses them. These support applications but do not name the general category of user-task software requested here.

Why each choice is right or wrong

A - Incorrect: The OS supplies resource management and services rather than naming the category of specific user-task programs.

B - Correct: Applications are explicitly described as software solving user tasks, matching the question’s requested category.

C - Incorrect: A programming language is a notation for programs, not the category of software performing user tasks.

D - Incorrect: A compiler specifically translates code; application software is the general user-task category requested.

Primary note proof

Study-note section COA02

An operating system manages resources and provides services.

Study-note section COA02

Applications solve user tasks.

Study-note section COA02

A compiler translates source into target code; assembly is a symbolic form of machine instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Computer Science MCQ practice collection · Question 730

Question 111 | AM0731

Which component provides main working storage for the program’s instructions and data, rather than the CPU’s immediate execution registers or persistent disk storage?

A. CPU

B. Main memory (RAM)

C. Hard disk

D. Motherboard

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Main memory (RAM)

Main memory supplies the addressed working storage that holds a program's instructions and data during operation. CPU registers retain particular fetched instructions, operands and results for immediate execution, while a hard disk supplies persistent storage. The revised stem and expanded memory label distinguish these levels rather than assuming all temporary storage belongs exclusively to one component. A motherboard provides physical interconnection rather than the requested storage role.

Why each choice is right or wrong

A - Incorrect: CPU registers serve immediate execution rather than the main working-storage level requested.

B - Correct: Main RAM is the addressed working store for program instructions and data.

C - Incorrect: A hard disk provides persistent backing storage rather than main working storage.

D - Incorrect: The motherboard physically connects components.

Primary note proof

Study-note section COA02

> In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

> 6.2 Main Memory
> The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.

• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science MCQ practice collection · Question 731

Question 112 | AM0732

Which type of software translates high-level programming languages into machine code?

A. Operating system

B. Application software

C. Programming language

D. Compiler

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Compiler

A compiler translates source code into target code; in this question the source is a high-level language and the target is machine code. The operating system manages resources, applications solve user tasks, and a programming language is the notation in which a program is expressed. The requested translation function therefore belongs to the compiler category.

Why each choice is right or wrong

A - Incorrect: An operating system manages resources and provides services; compiler is the translation category requested.

B - Incorrect: Application software solves user tasks rather than naming the general source-to-target translation category.

C - Incorrect: The passage distinguishes source high-level language from resulting machine language; a language itself is not the named translating agent (p.77).

D - Correct: Compiler support is explicitly tied to high-level-to-machine translation (p.77).

Primary note proof

Study-note section COA02

An operating system manages resources and provides services.

Study-note section COA02

Applications solve user tasks.

Study-note section COA02

A compiler translates source into target code; assembly is a symbolic form of machine instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Computer Science MCQ practice collection · Question 732

Question 113 | AM0733

The system clock in a computer is responsible for:

A. Controlling the execution speed of instructions

B. Storing data and instructions

C. Managing the input/output devices

D. Connecting different hardware components

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer A: Controlling the execution speed of instructions

Control selections are available at the beginning of a clock cycle and register data are captured on the next clock transition (PDF p.59). Fetch timing likewise advances after each clock pulse (p.55). Thus A is the closest description of clock timing instruction steps; it does not mean frequency alone determines total execution time.

Why each choice is right or wrong

A - Correct: Clock transitions time the operations and sequence steps (pp.55,59); this regulates execution pace.

B - Incorrect: Registers and memory store bits; the clock times their updates rather than stores the program (p.59).

C - Incorrect: I/O management is assigned to interfaces/control mechanisms, not solely the clock (pp.101-102).

D - Incorrect: Interconnection provides communication (p.6); the clock provides timing rather than that physical connection.

Primary note proof

Study-note section COA05

> Clocked sequential circuits update at designated clock events; combinational logic does not inherently require a clock.

Study-note section COA34

> CPU time=instruction_count×CPI×clock_period=instruction_count×CPI/clock_rate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 59 | lines 2-15

> The control unit that operates the CPU bus system directs the information flow through the register and ALU by selecting the various components in the system. For example, to perform the operation
> R1 R2 + R3, the control must provide binary selection variables to the following selector inputs:
> 1. MUX A selector (SELA): to place the content of R2 into bus A.
> 2. MUX selector (SELB): to place the content of R3 into bus B.
> 3. ALU operation selector (OPR): to provide the arithmetic add A +B.
> 4. Decoder destination selector (SELD): to transfer the content of output bus in to R1.
> The four control selection variables are generated in the control unit and must be available at the beginning of a clock cycle. The data from the two source registers propagates through the gates in the multiplexers and the ALU, to the output bus, and into the inputs of the destination register, all during the clock cycle interval. Then, when the next clock transition occurs, the binary information from the output bus is transferred into R1. To achieve a fast response

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be

brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 101 | lines 10-16

7.4 Input/output Problems
Wide variety of peripherals and delivering different amounts of data per second Work at different speeds Send/receive data in different formats all slower than CPU and RAM. Hence I/O modules are used as a solution.
Input/output Module: It is the entity within a computer that is responsible for the control of one or more external devices and for the exchange of data between those devices and main memory and/or CPU.

Question sources

Computer Science MCQ practice collection · Question 733

Question 114 | AM0734

Which process is responsible for fetching instructions from memory?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Fetching

The instruction-pipeline list explicitly begins “Fetch the instruction from memory” (PDF p.117), and the fetch micro-operations copy memory into IR (p.55). The question names this retrieval phase, not interpretation or computation.

Why each choice is right or wrong

A - Incorrect: Decoding interprets the retrieved instruction after fetching (p.117).

B - Incorrect: Execution carries out the operation later in the pipeline (p.117).

C - Correct: Fetching is the memory-to-instruction-register retrieval phase (pp.55,117).

D - Incorrect: Storing writes the result after execution, the opposite transfer purpose (p.117).

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

> Instruction Pipeline:
> Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
> 1. Fetch the instruction from memory
> 2. Decode the instruction
> 3. Calculate the effective address
> 4. Fetch the operands from memory
> 5. Execute the instruction
> 6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock

pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 70 | lines 17-24

The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science MCQ practice collection · Question 734

Question 115 | AM0735

Which process decodes the fetched instruction and determines the necessary actions?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Decoding

The module says decoding determines the operation, addressing mode and operand location (PDF p.70). It follows fetch and precedes execution in the instruction phases (p.117).

Why each choice is right or wrong

A - Correct: Decoding explicitly determines the required operation and addressing details.

B - Incorrect: Execution performs the operation after it has been determined (p.117).

C - Incorrect: Fetching retrieves the instruction bits before they are interpreted (p.117).

D - Incorrect: Storing writes the computed result, not the operation interpretation (p.117).

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

> Instruction Pipeline:
> Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
> 1. Fetch the instruction from memory
> 2. Decode the instruction
> 3. Calculate the effective address
> 4. Fetch the operands from memory

5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 55 | lines 2-8

1. Fetch an instruction from memory
2. Decode the instruction
3. Read the effective address from memory if the instruction has an indirect address
4. Execute the instruction
Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Question sources

Computer Science MCQ practice collection · Question 735

Question 116 | AM0736

Which process carries out the actual operations or calculations specified by the instruction?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Execution

The instruction stages separate fetch, decode, operand access, execution and result storage (PDF p.117). ALU supplies the required micro-operations for execution (p.57), so actual calculation is execution rather than any transport phase.

Why each choice is right or wrong

A - Incorrect: Decoding determines what to do, not the arithmetic itself.

B - Correct: Execution is the listed phase that performs the operation (pp.57,117).

C - Incorrect: Fetching retrieves the instruction and precedes the calculation (p.117).

D - Incorrect: Storing preserves the result after calculation (p.117).

Primary note proof

Study-note section COA15

A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

The control unit coordinates fetching and data flow; the ALU performs arithmetic and logic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division.

Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science MCQ practice collection · Question 736

Question 117 | AM0737

Which process is responsible for storing the results of the executed instruction?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Storing

The instruction-pipeline sequence ends by storing the result in its proper place (PDF p.117). The RISC variant specifies writing the result into a register (p.118), establishing the result-preservation role of D.

Why each choice is right or wrong

A - Incorrect: Decoding interprets operation and addressing fields, before a result exists.

B - Incorrect: Execution produces a result; the listed store phase preserves it afterward (p.117).

C - Incorrect: Fetching brings in the instruction, not its eventual result (p.117).

D - Correct: Store result/write register is the explicit final stage (pp.117-118).

Primary note proof

Study-note section COA15

A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

A write-back/result-storage phase is not another fetch.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 118 | lines 3-19

A RISC (Reduced Instruction Set Computer) processor pipeline operates in much the same way, although the stages in the pipeline are different. While different processors have different numbers of steps, they have basically variations of these five steps:
1. fetch instructions from memory
2. read registers and decode the instruction
3. execute the instruction or calculate an address
4. access an operand in data memory
5. write the result into a register
The length of the pipeline is dependent on the length of the longest step. Because RISC instructions are simpler than those used in pre-RISC processors (now called CISC, or Complex Instruction Set Computer), they are more conducive to pipelining. While CISC instructions varied in length, RISC instructions are all the same length and can be fetched in a single operation. Ideally, each of the stages in a RISC processor pipeline should take 1 clock cycle so that the processor finishes an instruction each clock cycle and averages one cycle per instruction (CPI). Among the characteristics attributed to RISC is its ability to use an efficient instruction pipeline.

Question sources

Computer Science MCQ practice collection · Question 737

Question 118 | AM0738

Which unit of the CPU is responsible for fetching instructions and data from memory?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Control Unit (CU)

The pipeline description explicitly says the control unit initiates the next fetch when there is buffer space (PDF p.117). CPU control supervises register transfers and ALU work (p.57); the fetch micro-operations transfer address and instruction bits (p.55). This is coordinating responsibility, not a claim the CU physically stores all fetched data.

Why each choice is right or wrong

A - Incorrect: ALU performs the arithmetic/logical micro-operations, not the fetch-sequencing role (p.57).

B - Correct: CU initiates fetch and supervises the register transfers (pp.57,117).

C - Incorrect: Cache supplies stored copies on hits (p.84); it does not decide the instruction-cycle control sequence.

D - Incorrect: Registers hold address/data/instruction values, while CU controls their transfers (pp.48,57).

Primary note proof

Study-note section COA15

> The control unit coordinates fetching and data flow; the ALU performs arithmetic and logic.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

> Instruction Pipeline:
> Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The

following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science MCQ practice collection · Question 738

Question 119 | AM0739

Which unit of the CPU performs arithmetic and logical operations?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Arithmetic Logic Unit (ALU)

ALU is explicitly the CPU part that performs required micro-operations; the module classifies arithmetic and logical manipulation as computational operations (PDF pp.57,74-75). Control selects the operation and registers supply/receive values.

Why each choice is right or wrong

A - Correct: ALU performs required arithmetic/logical micro-operations (p.57).

B - Incorrect: CU directs transfers and tells ALU which operation to perform (p.57).

C - Incorrect: Cache is a fast storage buffer (p.84), not the computation unit.

D - Incorrect: Register set stores intermediate data; ALU operates on it (p.57).

Primary note proof

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Study-note section COA01

> The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU

performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 739

Question 120 | AM0740

Which CPU storage class holds explicit execution operands and results, rather than copies of memory blocks?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Register Unit

Registers hold explicit execution operands and results. Cache also stores temporary information, but its role is retaining copies of memory blocks near the processor. The revised question uses that role distinction, so temporary storage alone no longer makes both choices correct. The ALU performs arithmetic and logic, and the control unit coordinates operations rather than being the requested operand-storage class.

Why each choice is right or wrong

A - Incorrect: The ALU computes rather than being the operand-storage class.

B - Incorrect: The control unit sequences operations rather than holding explicit execution operands as its defining role.

C - Incorrect: Cache holds copies of memory blocks.

D - Correct: Registers hold explicit execution operands and results.

Primary note proof

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA14

> The accumulator holds operands/results, SP tracks the stack, and status flags report conditions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions.

The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 740

Question 121 | AM0741

Which component of the CPU controls the flow of data and instructions within the CPU?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Control Unit (CU)

CU explicitly supervises information transfers among registers and tells ALU what operation to perform (PDF p.57). The bus-control example selects sources, operation and destination (p.59), matching control of internal flow.

Why each choice is right or wrong

A - Incorrect: ALU processes operands selected by control (pp.57,59), rather than directs the overall flow.

B - Correct: CU selects register sources, destination and ALU operation (p.59).

C - Incorrect: Cache stores instruction/data copies (p.84), not the CPU transfer-control sequence.

D - Incorrect: Registers hold intermediate data; CU controls their transfers (p.57).

Primary note proof

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Study-note section COA14

> A control word chooses ALU sources, destination and operation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the

computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 59 | lines 2-15

The control unit that operates the CPU bus system directs the information flow through the register and ALU by selecting the various components in the system. For example, to perform the operation
R1 R2 + R3, the control must provide binary selection variables to the following selector inputs:
1. MUX A selector (SELA): to place the content of R2 into bus A.
2. MUX selector (SELB): to place the content of R3 into bus B.
3. ALU operation selector (OPR): to provide the arithmetic add A +B.
4. Decoder destination selector (SELD): to transfer the content of output bus in to R1.
The four control selection variables are generated in the control unit and must be available at the beginning of a clock cycle. The data from the two source registers propagates through the gates in the multiplexers and the ALU, to the output bus, and into the inputs of the destination register, all during the clock cycle interval. Then, when the next clock transition occurs, the binary information from the output bus is transferred into R1. To achieve a fast response

Question sources

Computer Science MCQ practice collection · Question 741

Question 122 | AM0742

Which CPU-near storage component retains copies of memory blocks to speed repeated or nearby memory accesses?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Cache Unit

Cache retains copies of memory blocks near the CPU and exploits temporal and spatial locality. Registers instead hold explicit execution operands and results, even though frequently used values may also remain there. The ALU computes values and the control unit sequences operations. The memory-block-copy criterion therefore identifies cache without claiming that no other CPU component can hold useful data.

Why each choice is right or wrong

A - Incorrect: The ALU computes arithmetic and logic rather than retaining a memory-block-copy tier.

B - Incorrect: The control unit sequences transfers and operations rather than supplying this data-copy storage tier.

C - Correct: Cache stores nearby copies of memory blocks for repeated or spatially local accesses.

D - Incorrect: Registers hold explicit execution operands/results; that differs from the memory-block-copy cache role.

Primary note proof

Study-note section COA23

> Cache copies memory blocks near the CPU.

Study-note section COA23

> Temporal locality reuses recent data; spatial locality uses nearby locations.

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
> Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
> The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
> • The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
> • When CPU needs to access memory, the cache is examined
> • If the word is found in the cache, it is read from the fast memory
> • If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
> • When the CPU refers to memory and finds the word in cache, it is said to produce a hit
> • Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 85 | lines 2-8

• The performance of cache memory is frequently measured in terms of a quantity called hit ratio
Hit ratio = hit / (hit+miss)
• The basic characteristic of cache memory is its fast access time
• Therefore, very little or no time must be wasted when searching the words in the cache
• The transformation of data from main memory to cache memory is referred to as a mapping process.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science MCQ practice collection · Question 742

Question 123 | AM0743

During a taken branch instruction, which phase computes or applies the branch target to replace the sequential next-PC value?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Execution

A fetch uses the selected PC to obtain an instruction and can advance it to the sequential address. Executing a taken branch can later replace that sequential value with the branch target. The question now specifically asks about that branch action, rather than the broader process of selecting, reading or advancing a next instruction address. Decode interprets fields and write-back stores results; neither is the conceptual branch-execution role requested.

Why each choice is right or wrong

A - Incorrect: Decoding interprets opcode and addressing fields rather than performing the specified branch action.

B - Correct: Executing the taken branch replaces the sequential PC with its target.

C - Incorrect: Fetching obtains the instruction at the selected address rather than performing this branch action.

D - Incorrect: Result storage is not the conceptual branch-target execution phase.

Primary note proof

Study-note section COA15

> A branch can replace the sequential PC later.

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
> T0: AR ← PC
> Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
> T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
> II. Decode the instruction
> T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Question sources

Computer Science MCQ practice collection · Question 743

Question 124 | CS0232

What is the purpose of an opcode in a 16-bit instruction format (4-bit opcode | 6-bit operand ref | 6-bit operand ref)?

A. It holds the destination address of an operand

B. It determines the operation to be done

C. It holds the source address of an operand

D. It holds the next instruction reference

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: It determines the operation to be done

The stated field widths sum to 16 bits: 4+6+6. Their widths do not change the opcode's purpose: operation selection (PDF p.47). Operand reference fields identify source/destination values or locations; PC tracks sequence.

Why each choice is right or wrong

A - Incorrect: Destination reference is an operand/address field, not the opcode (p.47).

B - Correct: Operation selection is the explicit opcode definition (p.47).

C - Incorrect: Source reference belongs to the operand/address fields (p.47).

D - Incorrect: Next instruction is tracked by PC (p.70), not the opcode.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Study-note section COA16

> A 4-bit opcode permits at most 16 encodings, not necessarily 16 implemented instructions.

Study-note section COA16

> A 6-bit operand-reference field has 64 encodings; what they designate depends on the machine.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the

instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science Exit Exam practice collection · Question 232

Question 125 | CS0234

Which performance metric can be used to measure the time taken to perform a read or write operation?

A. Transfer rate

B. Memory cycle time

C. Fetching time

D. Access time

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Access time

Access time is the elapsed time from initiation to completion of a memory operation (PDF p.83). Memory cycle time is the minimum separation of successive operations. This distinguishes one read/write duration from transfer throughput or instruction fetching.

Why each choice is right or wrong

A - Incorrect: Transfer rate measures an amount per unit time; it is not the single-operation interval defined on p.83.

B - Incorrect: Cycle time measures spacing of successive operations, not the requested initiation-to-completion duration (p.83).

C - Incorrect: Fetching is one instruction-cycle task, not the general read/write performance metric (p.55).

D - Correct: Access time is exactly the operation-duration definition (p.83).

Primary note proof

Study-note section COA23

Access time measures latency of a read/write; cycle time is the minimum interval before another operation and may include recovery.

Study-note section COA23

Transfer rate measures data per time, not individual-request delay.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 83 | lines 11-14

The time that elapses between the initiation of an operation and completion of that operation is called memory access time. The minimum time delay between two successive memory operations is called memory cycle time. The cycle time is usually slightly lower than the access time.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

6.1 Memory Hierarchy:
A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1,
INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science Exit Exam practice collection · Question 234

Question 126 | CS0235

Under the textbook access-method taxonomy, how is magnetic-disk block access classified when requests identify blocks but mechanical positioning introduces delay?

A. Associative

B. Sequential

C. Random

D. Direct

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Direct

The stated textbook taxonomy calls magnetic disk access direct: a request identifies an addressed block, but positioning creates access delay. RAM is classified as random access with broadly address-independent timing, while tape is sequential and associative access matches content. Informal usage may call disks random-access devices too, but the revised question explicitly chooses the textbook classification convention rather than treating the labels as universally exclusive.

Why each choice is right or wrong

A - Incorrect: Associative access matches by content rather than addressed disk blocks.

B - Incorrect: Tape-like sequential access is not the disk's required access pattern.

C - Incorrect: Random is the RAM category in the explicitly stated textbook taxonomy.

D - Correct: Direct is the disk category in that taxonomy.

Primary note proof

Study-note section COA28

> In textbook access taxonomy, tape is sequential, disk is direct access with positioning delay, RAM is random access with broadly address-independent timing, and associative access matches content.

Study-note section COA28

> "Random-access disk" is also common informal usage; specify the taxonomy when comparing direct and random access.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Computer Organization and Architecture module | PDF page 98 | lines 7-16

6.4.2 Magnetic Tape
• Magnetic Tape is a plastic ribbon which is usually ½ inch or ¼ inch wide & 50 to 2400 feet long.
• It is coated with iron-oxide material.
• It is similar to the tape of audio cassettes of tape recorders. Data is stored as binary digits.
• Data is accessed sequentially so searching becomes difficult.
• Tape systems use the same reading and recording techniques as disk systems..
• Magnetic tape was the first kind of secondary memory.It is still widely used as the lowest-cost, slowest-speed member of the memory hierarchy.

Computer Organization and Architecture module | PDF page 87 | lines 2-15

Associative Mapping
• An associative mapping uses an associative memory.
• This memory is being accessed using its contents.
• Associative memory is expensive compared to RAM
• Each line of cache memory will accommodate the address (main memory) and the contents of that address from the main memory.
• That is why this memory is also called Content Addressable Memory (CAM). It allows each block of main memory to be stored in the cache.
A CPU address of 15 bits is places in the argument register and the associative memory us searched for a matching address
• If the address is found, the corresponding 12bits data is read and sent to the CPU
• If not, the main memory is accessed for the word
If the cache is full, an address-data pair must be displaced to make room for a pair that is needed and not presently in the cache

Question sources

Computer Science Exit Exam practice collection · Question 235

Question 127 | CS0236

Which statement reverses the broad integration trend described for computer evolution?

A. Increasing component size

B. Increasing processor speed

C. Increasing integration density

D. Increasing memory size

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer A: Increasing component size

The note describes a broad trend toward smaller individual components while density, capacity and processor performance increase. Increasing component size reverses that stated trend. The other alternatives describe growth in performance, integration density or memory capacity and are therefore consistent with it. These are historical tendencies, not guarantees that every later product improves every measured property.

Why each choice is right or wrong

A - Correct: The broad trend reduces individual component dimensions, so increasing size reverses it.

B - Incorrect: Rising processor performance is consistent with the stated evolutionary tendency.

C - Incorrect: Increasing integration density is explicitly part of the stated integration trend.

D - Incorrect: Growing storage capacity is consistent with the stated trend even as individual component dimensions shrink.

Primary note proof

Study-note section COA38

> Integration generally reduced individual component dimensions while capacity, density and processor performance grew; these are trends rather than guarantees for every model.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
> Figure 6.1 memory hierarchy

• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Computer Organization and Architecture module | PDF page 30 | lines 3-19

3.1 Integrated Circuits
• Integrated circuit (IC) is the basic building block of digital circuits.
• An integrated circuit is a small silicon semiconductor crystal, called a chip.
• The various gates are interconnected inside the chip to form the required circuit.
• As the technology of ICs has improved, the number of gates that can be put in a single chip has increased.
✓ Small-scale integration (SSI) devices contain several (usually less than 10) independent gates in a single package.
✓ Medium-scale integration (MSI) devices contain approximately 10 to 200 gates in a single package.
E.g To form decoders, adders, and registers.
✓ Large-scale integration (LSI) devices contain between 200 and a few thousands gates in a single package.
Eg. processors, memory chips, and programmable modules.
✓ Very-large-scale integration (VLSI) devices contain thousands of gates in a single

package.
E.g large memory arrays and complex microcomputer chips.

Question sources

Computer Science Exit Exam practice collection · Question 236

Question 128 | CS0237

Which one of the following is NOT correct about Von Neumann's architecture?

A. Data and instructions are stored separately

B. Modern computers are based on Von Neumann's architecture

C. Instruction execution is sequential

D. Memory content is located by its address

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Separate instruction and data storage contradicts the textbook Von Neumann model.

COA02 defines the textbook Von Neumann model as sharing an addressed memory system between instructions and data, with normally sequential execution unless control flow changes. Thus separate storage is the defining contradiction. The note allows modern machines to retain a unified address-space view while using split caches; ordinary execution and addressed memory remain compatible with that model.

Why each choice is right or wrong

A - Correct: Separate instruction/data memories describe Harvard organization rather than the textbook shared-memory Von Neumann model.

B - Incorrect: Modern systems can retain a unified address-space view, so this broad conventional association is compatible with the note.

C - Incorrect: Normally sequential instruction progression is part of the textbook model, with branches providing explicit control-flow changes.

D - Incorrect: The model explicitly uses addressed memory, so locating content by address is a compatible statement.

Primary note proof

Study-note section COA02

In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA02

Harvard separates instruction and data memories or paths.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 51 | lines 14-22

Instruction Set Completeness
A computer should have a set of instructions so that the user can construct machine language program to evaluate any function that is known to be computable. The set of instructions are said to be complete if the computer includes a sufficient number of instructions in each of the following categories:
1. Arithmetic, Logical and Shift Instructions.
2. Instructions for moving information to and from memory and processor registers.
3. Program control instructions together with instructions that check status conditions.
4. Input and Output instructions.

Question sources

Computer Science Exit Exam practice collection · Question 237

Question 129 | CS0238

Which instruction type is used to test the value of a data word or status of a condition?

A. Data processing instruction

B. Test instruction

C. Data movement instruction

D. Branch instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Test instruction.

COA19 states that TEST/CMP examine values and set condition information, often without retaining an arithmetic result. That exactly matches examining a data word or a condition status. Arithmetic/data processing is a broader computing class, data movement copies values, and branches choose control-flow targets. The named test-instruction category is therefore the direct answer among the options.

Why each choice is right or wrong

A - Incorrect: Data processing is a broader computing category; the stem specifically requests the instruction used to test values.

B - Correct: TEST/CMP examine values and establish condition information, directly matching the question's described operation.

C - Incorrect: Data movement copies values between locations rather than testing their contents or condition status.

D - Incorrect: Branches choose control-flow targets, often using previously established conditions rather than performing the requested test.

Primary note proof

Study-note section COA19

> TEST/CMP examine values and set condition information, often without retaining the arithmetic result.

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Study-note section COA19

> CALL saves a return point; RETURN restores it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

Data transfer instruction.
Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Computer Organization and Architecture module | PDF page 51 | lines 14-22

Instruction Set Completeness
A computer should have a set of instructions so that the user can construct machine language program to evaluate any function that is known to be computable. The set of instructions are said to be complete if the computer includes a sufficient number of instructions in each of the following categories:
1. Arithmetic, Logical and Shift Instructions.
2. Instructions for moving information to and from memory and processor registers.
3. Program control instructions together with instructions that check status conditions.
4. Input and Output instructions.

Question sources

Computer Science Exit Exam practice collection · Question 238

Question 130 | NS0221

What is the purpose of an opcode in a 16-bit instruction format (4-bit opcode | 6-bit operand ref 6-bit operand ref)?

A. It holds the destination address of an operand

B. It determines the operation to be done

C. It holds the source address of an operand

D. It holds the next instruction reference

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: The opcode specifies the operation.

COA16 defines the opcode as specifying the operation and operand fields as locating data/registers. In the given 16-bit format, the 4-bit opcode therefore selects an operation, while the two 6-bit references identify operands under the machine's interpretation. The bit widths are not evidence that the opcode stores a source/destination address or next-PC reference.

Why each choice is right or wrong

A - Incorrect: A destination operand reference belongs in an operand field, not in the operation-selecting opcode.

B - Correct: The opcode determines which operation the processor should perform for this instruction.

C - Incorrect: A source operand reference is an operand-field role rather than the opcode's operation role.

D - Incorrect: The next-instruction reference is a control-flow/PC concern, not the general purpose of this opcode field.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Question sources

Computer Science Exit Exam practice collection · Question 232

Question 131 | NS0289

A designer adds a PCI expansion card to a processor whose local bus uses a different transfer protocol. The two sides cannot be wired together directly. Which listed hardware category is needed to adapt transfers between the processor bus and the expansion interconnect?

A. SCSI

B. PCI

C. Bus bridge/controller logic

D. Multi bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Bus bridge/controller logic

The scenario requires hardware that adapts transfers between unlike bus segments, so bus bridge/controller logic is needed. PCI identifies the expansion interconnect on one side of the connection; merely naming it does not supply the adaptation to the processor bus. SCSI identifies a storage connection and Multibus another bus family. The distinction is between the interconnect being attached and the hardware that enables the attachment.

Why each choice is right or wrong

A - Incorrect: SCSI is a storage connection rather than the adapter between these unlike bus segments.

B - Incorrect: PCI names the expansion interconnect already present in the scenario, not the adapting component.

C - Correct: Bus bridge/controller logic adapts transfers between the unlike processor and expansion bus segments.

D - Incorrect: Multibus names a bus family rather than the component needed to adapt this connection.

Primary note proof

Study-note section COA32

> PCI is a peripheral expansion interconnect, not a disk protocol; connecting unlike bus segments requires suitable bridge/controller logic.

Question sources

Computer Science Exit Exam · 2018 · Question 6

Question 132 | NS0290

Combinational circuit differ from sequential circuit primarily because

A. Combinational circuit cannot be built using logic circuit

B. The output of combinational circuit depends only on the present input, not on past output.

C. Sequential circuit have no feedback while combinational circuit do

D. The combinational circuit requires clock signal, but sequential circuit do not

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: Combinational output depends on present inputs rather than stored history.

COA05 defines combinational output through present inputs after propagation delay, while sequential output also depends on stored state. B expresses that distinction. Combinational circuits are built from logic, not prohibited from it; the feedback statement reverses the usual state relationship. Nor do combinational circuits inherently require a clock, while clocked sequential circuits update at clock events.

Why each choice is right or wrong

A - Incorrect: Combinational circuits are logic circuits; the statement incorrectly denies their ordinary implementation basis.

B - Correct: Present inputs determine combinational outputs, whereas sequential circuits additionally depend on retained state.

C - Incorrect: This reverses the state/feedback distinction; sequential storage can involve feedback, not its categorical absence.

D - Incorrect: Combinational logic does not inherently require a clock; clock events characterize clocked sequential updates.

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Question sources

Computer Science Exit Exam · 2018 · Question 7

Question 133 | NS0304

Which one of the following is the first operation or step in CPU instruction cycle?

A. Executing the instruction

B. Handling interrupt

C. Decoding the instruction

D. Fetching the instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Fetch the instruction.

COA15 begins the conceptual instruction cycle with fetch, then decode, operand acquisition, execution and any result write-back. Fetch must obtain the instruction before its opcode can be decoded and operation performed. Permitted interrupt checks are a separate stage/transition in that conceptual model, not the first ordinary instruction-processing step among these choices.

Why each choice is right or wrong

A - Incorrect: Execution follows obtaining and decoding the instruction, rather than being the cycle's first conceptual step.

B - Incorrect: Interrupt checks/service are distinct from the first ordinary fetch step in the note's conceptual cycle.

C - Incorrect: Decoding requires an already fetched instruction, so it follows rather than precedes fetch.

D - Correct: Fetching obtains the instruction and is explicitly first in the conceptual instruction-cycle sequence.

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Question sources

Computer Science Exit Exam · 2018 · Question 22

Question 134 | NS0310

_____ is a data transfer method in computer organization and architecture that allows an I/O device to transfer data directly to or from memory without the involvement of the CPU.

A. Asynchronous data transfer

B. Direct Memory Access (DMA)

C. Interrupt-driven transfer

D. Programmed I/O transfer

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Direct Memory Access (DMA).

COA30 says DMA lets a controller transfer a block directly between device and memory after CPU setup. The stem's without CPU involvement must mean without CPU moving every item, not no configuration/completion involvement whatsoever. Polling and interrupt-driven I/O still require CPU transfer work; asynchronous transfer describes timing coordination rather than this block-moving mechanism.

Why each choice is right or wrong

A - Incorrect: Asynchronous transfer describes coordination/timing rather than specifically controller-managed movement directly between device and memory.

B - Correct: DMA transfers the block through a controller after CPU setup, avoiding per-item CPU copying.

C - Incorrect: Interrupt-driven handlers still consume CPU time to service and move device data in this comparison.

D - Incorrect: Programmed I/O repeatedly checks status and has the CPU move each item rather than using DMA.

Primary note proof

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Question sources

Computer Science Exit Exam · 2018 · Question 31

Question 135 | NS0367

Which logic gate produces output 1 only when ALL inputs are 1?

A. OR gate

B. AND gate

C. NAND gate

D. XOR gate

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer B: AND gate.

COA03 states that AND is 1 only when all inputs are 1. This is exactly the condition in the stem. OR needs any high input, NAND complements AND, and two-input XOR is high when inputs differ. Thus none of the other truth functions has the specified all-high-only output rule.

Why each choice is right or wrong

A - Incorrect: OR becomes high when any input is high, so it does not require all inputs to be one.

B - Correct: AND outputs one exactly for the all-inputs-one combination required by the stem.

C - Incorrect: NAND inverts AND and therefore outputs zero, not one, when all its inputs are one.

D - Incorrect: Two-input XOR outputs one when inputs differ, not when both inputs are one.

Primary note proof

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Question sources

Computer Organization & Architecture review questions · Question 1

Question 136 | NS0368

The Boolean expression F = A⊕B represents which gate?

A. XNOR gate

B. NOR gate

C. XOR gate

D. NAND gate

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: XOR gate.

The expression A⊕B is the exclusive-OR operation. COA03 defines its two-input output as 1 when the inputs differ and contrasts XNOR as agreement. NOR and NAND complement OR and AND instead. Thus XOR is the operation represented by the given symbol, with no extra circuit diagram or numeric assumption needed.

Why each choice is right or wrong

A - Incorrect: XNOR is the complement of XOR, giving one when the two inputs agree.

B - Incorrect: NOR complements OR; it is not the exclusive-OR operation denoted by the circled plus.

C - Correct: The circled-plus expression denotes exclusive OR, whose output is one for differing inputs.

D - Incorrect: NAND complements AND rather than implementing the given exclusive-OR expression.

Primary note proof

Study-note section COA-S06

> The circled plus symbol ⊕ denotes exclusive OR.

Study-note section COA03

> NAND complements AND; NOR complements OR.

Study-note section COA03

> For two inputs XOR is 1 when inputs differ, and XNOR when they agree.

Question sources

Computer Organization & Architecture review questions · Question 2

Question 137 | NS0369

Which gate is called the 'universal gate' because it can implement any Boolean function?

A. AND gate

B. OR gate

C. NAND gate

D. XOR gate

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: NAND, the only universal gate offered.

COA04 explicitly states that NAND and NOR are each universal and explains constructing NOT by tying inputs together, then building AND/OR combinations. Of those two universal families, only NAND is offered. AND and OR alone cannot provide inversion under this rule, and XOR is not identified as a universal family. Thus the correct offered gate is NAND.

Why each choice is right or wrong

A - Incorrect: AND alone is not the universal family described; it does not supply the required inversion construction.

B - Incorrect: OR alone is not the universal family described and cannot generate the stated NOT construction.

C - Correct: NAND is explicitly universal and can be combined to construct NOT, AND and OR.

D - Incorrect: XOR is not the NAND/NOR universal family; tied inputs produce zero rather than inversion.

Primary note proof

Study-note section COA04

> NAND and NOR are each universal: feeding the same input to both inputs constructs NOT; combine complemented gates to construct AND or OR.

Question sources

Computer Organization & Architecture review questions · Question 3

Question 138 | NS0370

For a NOR gate with inputs A=1, B=0, the output F is:

A. 0

B. 1

C. Undefined

D. Depends on voltage

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer A: F = 0.

COA03 defines OR as high when any input is high and NOR as its complement. At A=1, B=0, the OR value is 1, so NOR produces 0. The input combination is completely specified in ordinary Boolean logic; it is neither undefined nor dependent on an unspecified analog voltage model.

Why each choice is right or wrong

A - Correct: OR of 1 and 0 is 1; complementing that for NOR yields zero.

B - Incorrect: One is the OR result before inversion, not the NOR output for these inputs.

C - Incorrect: The Boolean NOR truth table fully defines this ordinary input combination.

D - Incorrect: The stem supplies Boolean input values, so no additional voltage threshold assumption is required.

Primary note proof

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Study-note section COA03

> NAND complements AND; NOR complements OR.

Question sources

Computer Organization & Architecture review questions · Question 4

Question 139 | NS0371

De Morgan's theorem states that NOT(A·B) equals:

A. NOT(A) · NOT(B)

B. NOT(A) + NOT(B)

C. A + B

D. NOT(A+B)

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer B: NOT(A·B) = NOT(A) + NOT(B).

COA04 gives (AB)'=A'+B'. Complementing a product changes AND to OR and complements each operand. B matches that exact transformation. A instead gives the complement of A+B, C omits the operand complements, and D leaves the complement attached to a sum, which represents NOR rather than the requested NAND function.

Why each choice is right or wrong

A - Incorrect: The product of complements equals NOT(A+B), rather than the complement of the original product.

B - Correct: Complement each operand and change AND to OR, exactly as De Morgan’s product rule states.

C - Incorrect: A+B omits both required complements and is not generally equal to NOT(A·B).

D - Incorrect: NOT(A+B) is NOR and differs from NOT(A·B), for example when only one input is high.

Primary note proof

Study-note section COA04

> De Morgan gives (AB)'=A'+B' and (A+B)'=A'B'.

Question sources

Computer Organization & Architecture review questions · Question 5

Question 140 | NS0372

In Boolean algebra, the expression A + A' (A OR NOT A) always equals:

A. 0

B. A

C. 1

D. A'

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: A + A' = 1.

COA04 explicitly lists A+A'=1. If A is 0 then A' is 1, and if A is 1 then A' is 0; either way their Boolean OR is 1. Neither A nor A' remains fixed across both assignments, and zero is the product-of-complements result rather than their sum.

Why each choice is right or wrong

A - Incorrect: Zero is A AND NOT A; the given operation is OR, so one input is always high.

B - Incorrect: A can be zero while the expression remains one, so the identity is not simply A.

C - Correct: A bit OR its complement is always one for both possible assignments.

D - Incorrect: A complement can be zero while the full expression is one, so it is not the constant result.

Primary note proof

Study-note section COA04

> Useful identities are A+0=A, A·1=A, A+A=A, AA=A, A+A'=1, AA'=0, A+AB=A and A(A+B)=A.

Question sources

Computer Organization & Architecture review questions · Question 6

Question 141 | NS0373

How many rows are in the truth table for a Boolean function with 4 input variables?

A. 8

B. 12

C. 16

D. 32

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: 16 rows.

COA03 gives 2^n rows for n binary inputs. For four independent variables, each has two assignments, so the total combinations are 2×2×2×2=16. The options 8 and 32 correspond to three and five binary variables respectively; 12 does not enumerate all four-variable combinations.

Why each choice is right or wrong

A - Incorrect: Eight is 2^3, the number of assignments for three inputs rather than four.

B - Incorrect: Twelve omits four of the sixteen possible combinations of four independent binary inputs.

C - Correct: Four independent binary inputs yield 2^4=16 distinct truth-table rows.

D - Incorrect: Thirty-two is 2^5 and would require five independent binary input variables.

Primary note proof

Study-note section COA03

> With n binary inputs, enumerate 2^n truth-table rows.

Question sources

Computer Organization & Architecture review questions · Question 7

Question 142 | NS0374

For the Boolean function F=x+y'z with x=0,y=1,z=1, which options evaluate to the same Boolean value as F under that assignment? Select all that apply.

A. 0

B. 1

C. x

D. z

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer A, C: 0; x

Complement first: y is one, so y-prime is zero. The product y-prime AND z is therefore zero, and OR with x, also zero, leaves F equal to zero. Both the literal zero option and the expression x evaluate to that result under the stated assignment. The literal one and z both evaluate to one. The multiple-selection wording preserves the actual option values rather than arbitrarily rejecting an equivalent expression.

Why each choice is right or wrong

A - Correct: The literal zero equals the computed function value.

B - Incorrect: One differs from the computed zero.

C - Correct: x is assigned zero and therefore equals the function value.

D - Incorrect: z is assigned one and differs from the function value.

Primary note proof

Study-note section COA03

> Evaluate complements, products and then sums.

Study-note section COA03

> A plus sign means Boolean OR, not integer addition.

Study-note section COA04

> Useful identities are A+0=A, A·1=A, A+A=A, AA=A, A+A'=1, AA'=0, A+AB=A and A(A+B)=A.

Question sources

Computer Organization & Architecture review questions · Question 8

Question 143 | NS0375

Which characteristic best describes a flip-flop?

A. A combinational logic circuit

B. A bistable multivibrator that stores one bit

C. A 4-bit counter

D. An arithmetic unit

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: A bistable one-bit storage element.

COA05 says a flip-flop stores one bit while powered and classifies flip-flops as sequential circuits with stored state. B is the option describing that storage element. A combinational circuit lacks stored logical state; a four-bit counter requires a multi-bit state structure, and an arithmetic unit computes rather than defining this one-bit storage component.

Why each choice is right or wrong

A - Incorrect: A combinational circuit has no stored logical state, whereas a flip-flop retains a bit.

B - Correct: Bistable one-bit storage matches the powered state-retention role of a flip-flop.

C - Incorrect: One flip-flop stores one bit; a four-bit counter requires a multi-bit state organization.

D - Incorrect: An arithmetic unit computes values rather than defining the one-bit sequential storage element.

Primary note proof

Study-note section COA-S07

> A flip-flop is bistable: it has two stable states representing 0 and 1.

Study-note section COA-S07

> Multiple flip-flops are required to represent a multi-bit counter state.

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Question sources

Computer Organization & Architecture review questions · Question 9

Question 144 | NS0376

What distinguishes a flip-flop from a simple logic gate in terms of memory?

A. Flip-flops use more power

B. Flip-flops retain state even after inputs change

C. Flip-flops operate faster

D. Flip-flops require AC power

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: Flip-flops retain stored state during powered operation.

COA05 distinguishes sequential state from combinational outputs determined by present inputs. A flip-flop can retain its bit when inputs change without a qualifying update, giving the memory distinction in B. This does not imply nonvolatile retention after power-off. Power, speed and AC-supply claims are neither the defining property nor guarantees established here.

Why each choice is right or wrong

A - Incorrect: Power consumption is implementation-dependent and does not define whether a circuit has logical memory.

B - Correct: Stored state persists between valid updates while powered, unlike a purely present-input combinational gate.

C - Incorrect: Speed is not the defining memory distinction and is not guaranteed by the flip-flop label.

D - Incorrect: The note specifies powered storage, not a requirement for an AC rather than another supply.

Primary note proof

Study-note section COA05

> A flip-flop stores one bit while powered.

Study-note section COA05

> State retention during operation does not mean nonvolatile storage.

Question sources

Computer Organization & Architecture review questions · Question 10

Question 145 | NS0377

What is the decimal equivalent of binary number 1011?

A. 9

B. 10

C. 11

D. 13

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer C: $1011_2 = 11_{10}$.

Using COA08's base-position rule, 1011_2 has weights 8, 4, 2 and 1. Its high bits contribute 8+2+1=11, while the 4-place is zero. The note's worked example confirms the same sum. Nine, ten and thirteen correspond to different low-order bit patterns, not the supplied 1011.

Why each choice is right or wrong

A - Incorrect: Nine equals 8+1, corresponding to 1001_2 and omitting the given two-place bit.

B - Incorrect: Ten equals 8+2, corresponding to 1010_2 and omitting the given one-place bit.

C - Correct: The set bit positions contribute 8+2+1=11 in decimal.

D - Incorrect: Thirteen equals 8+4+1, corresponding to 1101_2 rather than the supplied 1011_2.

Primary note proof

Study-note section COA08

> In base b, a digit at position i has value digit×b^i.

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Question sources

Computer Organization & Architecture review questions · Question 11

Question 146 | NS0378

In hexadecimal, what does the letter 'F' represent in decimal?

A. 14

B. 15

C. 16

D. 10

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer B: $F_{16} = 15_{10}$.

COA08 defines hexadecimal digits as 0-9,A-F and explicitly gives F=15. These sixteen symbols represent values zero through fifteen, so sixteen itself requires the two-digit hexadecimal representation 10. Fourteen is E, and ten is A, leaving fifteen as the only value of the single digit F.

Why each choice is right or wrong

A - Incorrect: Fourteen corresponds to the preceding hexadecimal digit E, not the final digit F.

B - Correct: The note explicitly assigns decimal value fifteen to the hexadecimal digit F.

C - Incorrect: Sixteen requires hexadecimal 10; it is outside the value range of a single hexadecimal digit.

D - Incorrect: Ten corresponds to hexadecimal A rather than the later digit F.

Primary note proof

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Question sources

Computer Organization & Architecture review questions · Question 12

Question 147 | NS0379

The octal number system uses a base of:

A. 2

B. 8

C. 10

D. 16

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer B: Base 8.

COA08 gives octal digits 0-7: eight possible digit values, so octal has base eight. Binary has two digits, decimal ten and hexadecimal sixteen. Place weights in octal are therefore powers of eight, not powers of two, ten or sixteen, even though octal digits can be grouped with three binary bits.

Why each choice is right or wrong

A - Incorrect: Base two is binary; octal allows eight digit values rather than only zero and one.

B - Correct: Octal uses digits zero through seven, giving a base of eight.

C - Incorrect: Base ten is decimal, not the eight-digit octal number system.

D - Incorrect: Base sixteen is hexadecimal, which uses the additional A-F digits absent from octal.

Primary note proof

Study-note section COA08

> In base b, a digit at position i has value digit×b^i.

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Question sources

Computer Organization & Architecture review questions · Question 13

Question 148 | NS0380

Which number system is most fundamental to digital computing and why?

A. Decimal, because humans use it

B. Octal, because it maps to 3 bits

C. Binary, because digital circuits have two states (on/off)

D. Hexadecimal, because it is compact

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: Binary matches the two logical states used by digital logic.

COA03 defines digital gates using the two Boolean values 0 and 1, and COA08 defines binary using exactly those two digits. That direct correspondence explains why binary is fundamental to the digital representation described here. Octal and hexadecimal group binary bits for convenience, while decimal's human familiarity does not supply the two-state circuit correspondence.

Why each choice is right or wrong

A - Incorrect: Human decimal familiarity does not match the two Boolean states used here's digital gates.

B - Incorrect: Octal groups three binary bits and is a convenient notation rather than the underlying two-state representation.

C - Correct: Binary digits zero and one directly correspond to the two logical values used by digital circuits.

D - Incorrect: Hexadecimal compactly groups four binary bits; compact notation does not replace the underlying binary states.

Primary note proof

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Study-note section COA03

> NOT complements a bit.

Question sources

Computer Organization & Architecture review questions · Question 14

Question 149 | NS0381

How many bits are needed to represent the hexadecimal digit F?

A. 2

B. 3

C. 4

D. 8

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer C: 4 bits: $F_{16} = 1111_2$.

COA08 groups four binary bits for each hexadecimal digit and gives F=15. COA09's unsigned range shows three bits reach only 7, while four bits reach 15. Therefore the minimum representation of F is four bits, 1111. Eight bits can contain the value but are not the number needed for one hexadecimal digit.

Why each choice is right or wrong

A - Incorrect: Two bits encode only zero through three, insufficient for hexadecimal F's decimal value fifteen.

B - Incorrect: Three bits encode only zero through seven, so they cannot represent the value fifteen.

C - Correct: Four bits encode zero through fifteen; F is represented by the four-bit pattern 1111.

D - Incorrect: Eight bits could contain fifteen but exceed the minimum four bits needed for a hexadecimal digit.

Primary note proof

Study-note section COA08

> Group binary bits in threes for octal and fours for hexadecimal, outward from the radix point.

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Study-note section COA09

> For n-bit unsigned values the range is 0 through 2^n−1.

Question sources

Computer Organization & Architecture review questions · Question 15

Question 150 | NS0382

Which listed memory categories lose their contents when power is removed? Assume ordinary SRAM/DRAM RAM and an SRAM-based processor cache. Select all that apply.

A. ROM

B. Flash memory

C. RAM

D. Cache

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C, D: RAM; Cache

Ordinary SRAM and DRAM require power to retain their contents. RAM therefore qualifies, and the stated SRAM-based cache qualifies for the same technological reason. Cache is a hierarchy role rather than a promise of nonvolatile storage. ROM and flash retain contents without power, so they are excluded. Specifying implementation makes the volatility judgment definite while keeping both valid categories rather than forcing a RAM-only answer.

Why each choice is right or wrong

A - Incorrect: ROM retains contents without power.

B - Incorrect: Flash retains contents without power.

C - Correct: Ordinary SRAM/DRAM RAM is volatile.

D - Correct: The explicitly SRAM-based cache is volatile too.

Primary note proof

Study-note section COA24

> SRAM uses powered bistable storage: fast and costly per bit, without periodic refresh.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA24

> Both are ordinarily volatile.

Study-note section COA24

> ROM and flash retain contents without power; erasable/programmed variants mean “ROM can never be modified” is too broad.

Question sources

Computer Organization & Architecture review questions · Question 16

Question 151 | NS0383

Cache memory is best described as:

A. Large, slow, cheap memory

B. Small, ultra-fast memory between CPU and RAM

C. Non-volatile backup storage

D. Virtual memory on disk

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Small, fast storage near the CPU between it and main memory.

COA23 places cache above DRAM in the usual speed/cost hierarchy and says faster levels are generally smaller and more expensive per bit. It defines cache as copied memory blocks near the CPU. B captures that role. Cache is not the large slow backup layer or disk-based virtual-memory backing store; its useful speed derives from proximity/reuse, not guaranteed persistence.

Why each choice is right or wrong

A - Incorrect: Large slow cheap storage describes lower hierarchy levels rather than the small fast cache layer.

B - Correct: Cache is a small fast near-CPU copy of memory blocks used to accelerate access.

C - Incorrect: The note defines cache for accelerated reuse, not as nonvolatile backup storage.

D - Incorrect: Disk-backed virtual memory is a different address-space/backing-store mechanism from the CPU cache.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA23

> Faster levels are generally smaller and more expensive per bit.

Study-note section COA23

> Cache copies memory blocks near the CPU.

Question sources

Computer Organization & Architecture review questions · Question 17

Question 152 | NS0384

In direct memory mapping, what happens when two different main memory blocks map to the same cache slot?

A. Both are stored simultaneously

B. A cache conflict occurs and one block replaces the other

C. The system crashes

D. The blocks merge

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: A placement conflict: only one block can occupy that direct-mapped slot.

COA25 states that two blocks with the same direct-mapped index cannot reside together. When the requested block differs from the valid block at that slot, the cache must replace the slot's contents, producing a conflict miss. This is ordinary cache operation, not a crash or merging of block contents; direct mapping offers no alternate placement for the same index.

Why each choice is right or wrong

A - Incorrect: One direct-mapped line holds one block at a time, so the colliding blocks cannot coexist there.

B - Correct: The shared index causes a placement conflict; installing the requested block replaces the current one.

C - Incorrect: Conflict replacement is ordinary cache behavior and does not imply a system crash.

D - Incorrect: Cache replacement preserves separate memory-block identities; it does not merge their contents into one block.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Two blocks with the same direct-mapped index cannot reside together.

Question sources

Computer Organization & Architecture review questions · Question 18

Question 153 | NS0385

Which cache mapping technique offers the most flexibility in placing data blocks?

A. Direct mapping

B. Set associative mapping

C. Fully associative mapping

D. Sequential mapping

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Fully associative mapping

COA25 says direct mapping fixes one line, set associativity permits any way within one indexed set, and fully associative placement permits any line. Allowing every line gives the most placement flexibility for a fixed cache. The sequential-mapping option is not one of the cache placement rules defined by the supplied note.

Why each choice is right or wrong

A - Incorrect: Direct mapping restricts a block to a single line and therefore has the least placement freedom.

B - Incorrect: Set associativity permits several ways in one set, fewer locations than the entire fully associative cache.

C - Correct: Fully associative placement allows any cache line, maximizing placement freedom for each block.

D - Incorrect: Sequential ordering is not a placement freedom comparable to the defined direct, set-associative and fully associative cache rules.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Fully associative placement permits any line.

Study-note section COA25

> A W-way set-associative cache has S=L/W sets and set=block mod S; any of W ways in that set may hold the block.

Question sources

Computer Organization & Architecture review questions · Question 19

Question 154 | NS0386

Set associative mapping is described as a compromise between:

A. RAM and ROM

B. Direct and fully associative mapping

C. Primary and secondary memory

D. Cache and virtual memory

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Direct and fully associative mapping.

COA25 defines W-way set associativity as fixing a set by the block index while allowing any of W ways inside it. That combines direct indexing with an associative choice within the selected set, placing it between one-line direct mapping and unrestricted fully associative placement. RAM/ROM and primary/secondary are technology/hierarchy contrasts, not the two placement endpoints.

Why each choice is right or wrong

A - Incorrect: RAM and ROM describe memory types rather than the placement-policy endpoints combined by set associativity.

B - Correct: Set indexing restricts placement like direct mapping while allowing an associative choice among the selected set's ways.

C - Incorrect: Primary and secondary describe hierarchy roles, not the direct versus associative cache-placement alternatives.

D - Incorrect: Cache and virtual memory are different mechanisms rather than the two placement extremes defining set associativity.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Fully associative placement permits any line.

Study-note section COA25

> A W-way set-associative cache has S=L/W sets and set=block mod S; any of W ways in that set may hold the block.

Question sources

Computer Organization & Architecture review questions · Question 20

Question 155 | NS0387

Virtual memory primarily uses which storage device to extend the apparent memory size?

A. ROM

B. Cache

C. Hard disk / SSD

D. Registers

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Hard disk / SSD as backing storage.

COA27 says backing storage can hold nonresident pages while virtual memory provides translated protected address spaces. COA28 identifies HDDs and SSDs as persistent storage devices, matching that backing role. ROM, cache and registers are not the nonresident-page backing tier described. Apparent extension does not mean physical RAM is added or that virtual memory is only disk disguised as RAM.

Why each choice is right or wrong

A - Incorrect: ROM stores persistent firmware/data but is not the writable nonresident-page backing role described here.

B - Incorrect: Cache holds near-CPU copies of memory blocks rather than serving as the mass backing store.

C - Correct: Persistent HDD/SSD storage can hold nonresident page contents for the virtual-memory system.

D - Incorrect: Registers hold immediate execution state and cannot serve as the large backing store for nonresident pages.

Primary note proof

Study-note section COA27

> Virtual memory gives processes translated address spaces and protection; backing storage can hold nonresident pages.

Study-note section COA28

> An HDD stores magnetic patterns on tracks divided into sectors; reads/writes access addressed blocks while platters rotate.

Study-note section COA28

> SSDs use nonvolatile semiconductor storage without mechanical seek.

Question sources

Computer Organization & Architecture review questions · Question 21

Question 156 | NS0388

A valid disk-backed virtual page is currently nonresident, and the access must load its saved contents before continuing. Which condition describes the resulting page-fault case?

A. The CPU runs out of registers

B. The requested page is not in physical memory and must be loaded from disk

C. Cache memory is full

D. An arithmetic overflow occurs

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: The requested page is not in physical memory and must be loaded from disk

Accessing the valid but nonresident disk-backed page faults so that the system can make the page resident and continue. The stated case requires loading its saved contents. Register pressure, cache replacement and arithmetic overflow are different conditions. Other kinds of page fault need not read a disk, so the question explicitly limits itself to this nonresident disk-backed case.

Why each choice is right or wrong

A - Incorrect: Register allocation pressure is not an access to a missing or forbidden virtual page.

B - Correct: The described disk-backed nonresident access is a page-fault case, although not all faults require a disk read.

C - Incorrect: A full cache leads to cache replacement, not necessarily a virtual-memory page fault.

D - Incorrect: Arithmetic overflow is an instruction arithmetic condition rather than a virtual-page access event.

Primary note proof

Study-note section COA27

> A valid nonresident page access faults and may require loading data; invalid or forbidden accesses can also fault.

Study-note section COA27

> A cache miss, TLB miss and page fault are different events; not every page fault reads a disk, since zero-fill and already-resident shared pages exist.

Question sources

Computer Organization & Architecture review questions · Question 22

Question 157 | NS0389

Compared to RAM, a Hard Disk Drive (HDD) is:

A. Faster and volatile

B. Slower and non-volatile

C. Faster and non-volatile

D. Slower and volatile

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: HDD is generally slower and nonvolatile.

COA23 places DRAM above HDD in the usual speed hierarchy. COA28 says an HDD does not require continuous power to preserve stored bits, while COA24 calls ordinary SRAM/DRAM volatile. Combining the independent speed and retention rules gives slower and nonvolatile. The answer concerns the usual hierarchy comparison rather than every conceivable workload or specialized memory device.

Why each choice is right or wrong

A - Incorrect: It reverses both usual properties: HDD is slower than RAM and retains data without power.

B - Correct: The hierarchy places HDD below RAM in speed, and magnetic storage preserves bits without continuous power.

C - Incorrect: Nonvolatility is right, but the claim that HDD is faster than RAM reverses the usual hierarchy.

D - Incorrect: Slower is right, but volatile contradicts the note's explicit HDD power-off retention statement.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA24

> Both are ordinarily volatile.

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Question sources

Computer Organization & Architecture review questions · Question 23

Question 158 | NS0390

Which range matches the historical optical-disk access-time examples used in this course, rather than a universal specification for every optical drive?

A. 1-5 milliseconds

B. 10-30 milliseconds

C. 100-300 milliseconds

D. 500-1000 milliseconds

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: 100-300 milliseconds

The note identifies one hundred to three hundred milliseconds as a historical optical-access example. That range is not a universal specification: actual timing depends on the device and measurement conditions. The revised stem asks for the particular teaching range rather than an unqualified typical value for all optical disks. The remaining intervals do not match that historical example and therefore are definite distractors.

Why each choice is right or wrong

A - Incorrect: This interval does not match the historical optical-access example.

B - Incorrect: This shorter interval is not the course's cited historical example.

C - Correct: This is the historical example range given in the note.

D - Incorrect: This longer interval is not the course's cited historical example.

Primary note proof

Study-note section COA28

> Optical access ranges such as 100-300 ms are historical examples, not universal device specifications.

Question sources

Computer Organization & Architecture review questions · Question 24

Question 159 | NS0391

Which storage medium accesses data sequentially rather than randomly?

A. SSD

B. HDD

C. Magnetic tape

D. Optical disk

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Magnetic tape.

COA28 classifies tape as sequential and suited to streaming/archives. Disk provides direct addressed-block access, while SSD provides nonmechanical persistent storage rather than a tape-style stream. The question asks the medium whose access is characteristically sequential, so magnetic tape is the supported choice. Physical streaming inside another device does not make its logical access interface tape-like.

Why each choice is right or wrong

A - Incorrect: SSD is addressed nonvolatile semiconductor storage rather than the sequential tape-access medium described.

B - Incorrect: HDD permits addressed-block direct access rather than requiring tape-like sequential record traversal.

C - Correct: Tape is explicitly the sequential streaming/archive medium in the note's access taxonomy.

D - Incorrect: Optical disk uses addressed disk access rather than being the sequential tape medium requested.

Primary note proof

Study-note section COA28

> Optical media use lasers; tape is suited to sequential streaming and archives.

Study-note section COA28

> In textbook access taxonomy, tape is sequential, disk is direct access with positioning delay, RAM is random access with broadly address-independent timing, and associative access matches content.

Question sources

Computer Organization & Architecture review questions · Question 25

Question 160 | NS0392

Registers within the CPU are used primarily to:

A. Store large datasets permanently

B. Hold temporary data for immediate CPU operations

C. Communicate with I/O devices

D. Manage virtual memory pages

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Hold temporary values for immediate CPU operations.

COA14 distinguishes registers as explicit execution operands and assigns them fetched instructions, transferred data and operands/results. That supports the immediate temporary-storage role in B. Registers are small execution state rather than permanent bulk data storage. Some registers participate in I/O or translation, but those specialized uses do not replace the primary execution-storage role asked here.

Why each choice is right or wrong

A - Incorrect: Registers are small temporary execution storage rather than permanent bulk storage for large datasets.

B - Correct: Registers hold operands, instruction state and intermediate results used directly by current CPU operations.

C - Incorrect: Some registers support I/O, but general I/O communication is not their primary execution-storage purpose.

D - Incorrect: Translation can use specialized state, but managing virtual pages is not the primary role of CPU registers generally.

Primary note proof

Study-note section COA14

> IR holds the fetched instruction; MAR holds an address used for the current memory access; MDR holds transferred data.

Study-note section COA14

> The accumulator holds operands/results, SP tracks the stack, and status flags report conditions.

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Question sources

Computer Organization & Architecture review questions · Question 26

Question 161 | NS0393

The address bus in a computer system is responsible for:

A. Carrying actual data between components

B. Specifying the memory location the CPU wants to access

C. Providing timing signals

D. Transmitting control commands

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Specify the addressed memory location.

COA12 says address signals select a location, data signals carry values and control signals coordinate read/write and timing. Therefore B directly matches the address bus. Actual payload data belongs to the data bus, while timing and operation commands belong to control signaling. The distinction does not depend on address width or the value being transferred.

Why each choice is right or wrong

A - Incorrect: Actual payload values travel on the data bus rather than the address signals selecting a location.

B - Correct: The address bus identifies the memory location selected for the CPU's access.

C - Incorrect: Timing belongs to control signaling, not the location-selection role of address signals.

D - Incorrect: Read/write and other operation commands belong to control signaling rather than the address bus.

Primary note proof

Study-note section COA12

> Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Question sources

Computer Organization & Architecture review questions · Question 27

Question 162 | NS0394

A CPU exposes 32 address bits and uses byte-addressable memory with no additional address-extension mechanism. What maximum byte capacity can it address? Use GB here to mean 2^30 bytes.

A. 2 GB

B. 4 GB

C. 8 GB

D. 16 GB

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: 4 GB

Thirty-two address bits identify 2^32 distinct locations. With one byte per location, that is 2^32 bytes, or four units of 2^30 bytes. The selected option is therefore four GB under the explicitly stated binary convention, more precisely four GiB. Data-bus width is irrelevant to this location count, and without byte addressing the same number of address bits would not determine a byte capacity.

Why each choice is right or wrong

A - Incorrect: Two binary gigabytes would require only 2^31 byte locations.

B - Correct: 2^32 byte locations equal four units of 2^30 bytes.

C - Incorrect: Eight binary gigabytes would require 33 address bits.

D - Incorrect: Sixteen binary gigabytes would require 34 address bits.

Primary note proof

Study-note section COA12

> With a address bits there are 2^a addressable locations.

Study-note section COA12

> Multiply by bytes per location before calling this a byte capacity.

Question sources

Computer Organization & Architecture review questions · Question 28

Question 163 | NS0395

The control bus in a computer system carries:

A. Data between CPU and memory

B. Memory addresses

C. Signals that regulate operations like read/write timing

D. Interrupt requests only

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Control signals coordinate operations, including read/write timing.

COA12 defines control signals as coordinating read/write, timing, requests and acknowledgements. C expresses that general role. The data and address alternatives belong to distinct signal groups. Interrupt requests may be part of aggregate control, but the word only makes D too restrictive because read/write and timing controls are also included.

Why each choice is right or wrong

A - Incorrect: Actual transferred data belongs to the data bus rather than the control-signal group.

B - Incorrect: Location selection belongs to address signals rather than the control bus's coordination role.

C - Correct: Read/write coordination and timing are explicitly among the functions assigned to control signals.

D - Incorrect: Interrupt requests are only one possible control input; the bus also carries other coordination signals.

Primary note proof

Study-note section COA12

> Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Study-note section COA12

> Aggregate control signals include both CPU outputs and device inputs; both data and control buses can therefore be bidirectional under that scope.

Question sources

Computer Organization & Architecture review questions · Question 29

Question 164 | NS0396

In instruction format terminology, an 'opcode' refers to:

A. The memory address of the next instruction

B. The bits specifying the operation to perform

C. The register holding the result

D. The data value being processed

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Bits specifying the operation.

COA16 defines the opcode as the operation-specifying field. Operand fields locate registers/data and mode fields explain their interpretation. Thus B names the correct format role; next-instruction addresses, result-register identity and operand values are separate quantities. Their presence elsewhere in a machine does not change what opcode means.

Why each choice is right or wrong

A - Incorrect: Next-instruction position is a PC/control-flow role, not the opcode's operation-selection role.

B - Correct: Opcode bits select the operation the processor should perform for the instruction.

C - Incorrect: Result-register selection belongs to an operand/destination field rather than defining the opcode.

D - Incorrect: An operand data value is distinct from the code selecting the operation to apply.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Question sources

Computer Organization & Architecture review questions · Question 30

Question 165 | NS0397

A zero-address instruction format relies on:

A. Implicit register addressing

B. Stack-based operand access

C. Immediate values only

D. Direct memory addressing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Implicit top-of-stack operands.

COA16 says zero-address stack ADD uses two implicit top-of-stack operands. The absence of written operand addresses therefore relies on stack organization, not immediate constants or direct memory fields. Generic implicit register addressing is too broad and does not identify the stack discipline specified for this format; PUSH/POP may still have explicit addresses.

Why each choice is right or wrong

A - Incorrect: Implicit register addressing alone does not identify the stack operand convention defining zero-address arithmetic here.

B - Correct: Operands come implicitly from the top of the stack, so arithmetic needs no explicit operand-address fields.

C - Incorrect: Immediate values require encoded operand information; zero-address stack arithmetic is not restricted to constants.

D - Incorrect: Direct memory addressing would use an explicit operand address, contrary to the stated zero-address arithmetic format.

Primary note proof

Study-note section COA16

> Zero-address stack ADD uses two implicit top-of-stack operands.

Study-note section COA17

> PUSH and POP may still contain an explicit address even on a stack machine.

Question sources

Computer Organization & Architecture review questions · Question 31

Question 166 | NS0398

Which instruction format is most expressive, allowing operations on three separate operands?

A. Zero-address

B. One-address

C. Two-address

D. Three-address

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Three-address format.

COA16 defines three-address ADD X,Y,Z as X←Y+Z, allowing a separately named destination and two sources. This matches the stem's three separate operands. Two-address form overwrites one source, one-address uses an implicit accumulator, and zero-address uses stack operands. Most expressive is interpreted here through the explicit three-operand criterion rather than a universal comparison of complete instruction sets.

Why each choice is right or wrong

A - Incorrect: Zero-address arithmetic names no explicit operands and instead uses the implicit stack top.

B - Incorrect: One-address accumulator arithmetic names only one explicit operand and relies on an implicit accumulator.

C - Incorrect: Two-address arithmetic overwrites a named source and lacks a separately named third operand field.

D - Correct: Three explicit fields allow one destination and two independently named source operands.

Primary note proof

Study-note section COA16

> Three-address ADD X,Y,Z means X←Y+Z under destination-first notation.

Study-note section COA16

> Two-address ADD X,Y overwrites X with X+Y.

Study-note section COA16

> One-address accumulator ADD X means AC←AC+M[X].

Study-note section COA16

> Zero-address stack ADD uses two implicit top-of-stack operands.

Question sources

Computer Organization & Architecture review questions · Question 32

Question 167 | NS0399

Stack organization follows which principle?

A. FIFO (First In First Out)

B. LIFO (Last In First Out)

C. Random Access

D. Priority-based Access

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: LIFO: last in, first out.

COA17 defines a stack as last-in, first-out: PUSH adds at the logical top and POP removes that top item. The most recently pushed remaining item is therefore the next removed. FIFO describes queue ordering, while arbitrary indexed access or priority selection does not define the stack discipline.

Why each choice is right or wrong

A - Incorrect: FIFO removes the oldest inserted item, the opposite order from normal stack removal.

B - Correct: The last item pushed becomes the top and is the first item popped.

C - Incorrect: Random access does not enforce removal of the most recently pushed remaining item.

D - Incorrect: Stack removal follows insertion order in reverse, not a priority ranking assigned to items.

Primary note proof

Study-note section COA17

> A stack is last-in, first-out.

Study-note section COA17

> PUSH adds an item and POP removes the logical top.

Question sources

Computer Organization & Architecture review questions · Question 33

Question 168 | NS0400

In indirect addressing mode, the instruction contains:

A. The actual data value

B. The direct memory address of the data

C. A pointer to the memory address where data resides

D. A register number

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: A reference whose stored contents supply the effective address.

COA18 defines memory-indirect addressing as EA=M[A], followed by operand M[EA]. Thus the instruction's field identifies a pointer location rather than directly containing the operand value or final operand address. C is the pointer/indirection description. A describes immediate, B direct, and a plain register number identifies a register-based mode rather than this memory-indirect convention.

Why each choice is right or wrong

A - Incorrect: An actual value encoded in the instruction is immediate addressing rather than memory indirection.

B - Incorrect: Using the field directly as the operand's address is direct addressing, without the intermediate pointer lookup.

C - Correct: The field identifies a pointer-bearing location; its contents supply the address of the operand data.

D - Incorrect: A register number names register-based addressing rather than the memory-indirect field defined here.

Primary note proof

Study-note section COA18

> Let A be an instruction field, R a register value and M[x] the contents at x.

Study-note section COA18

> Immediate uses value A; direct uses EA=A and operand M[A]; indirect uses EA=M[A] then operand M[EA].

Study-note section COA18

> Register uses operand R; register-indirect uses EA=R.

Question sources

Computer Organization & Architecture review questions · Question 34

Question 169 | NS0401

Which is a textbook RISC design tendency, without claiming every instruction has single-cycle latency?

A. Variable-length instructions

B. Large instruction set (100-250 ops)

C. Simple operations, regular encodings and load/store memory access

D. Complex microcode control unit

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: Simple operations, regular encodings and load/store memory access

Textbook RISC emphasizes simple operations, regular encodings, load/store memory access and pipeline-friendly control. These tendencies do not imply every instruction completes with one-cycle latency; steady-state pipeline throughput is a separate measure. Variable lengths and complex operations are associated with CISC tendencies, while an instruction-count range is not a defining guarantee. The choice states these RISC characteristics without imposing a universal single-cycle latency.

Why each choice is right or wrong

A - Incorrect: Variable lengths are associated with CISC tendencies rather than the stated RISC summary.

B - Incorrect: This numerical instruction-count range is not the supported RISC characteristic.

C - Correct: These are the RISC design tendencies named in the note.

D - Incorrect: Complex microcode control is not the stated RISC design tendency.

Primary note proof

Study-note section COA20

> Textbook RISC emphasizes simple operations, regular encodings, load/store memory access, plentiful registers and pipeline-friendly control.

Study-note section COA20

> CISC commonly supplies more complex operations, richer addressing modes and variable lengths.

Study-note section COA20

> One instruction completing per pipeline cycle is throughput, not proof that every instruction has one-cycle latency.

Question sources

Computer Organization & Architecture review questions · Question 35

Question 170 | NS0402

CISC processors differ from RISC processors primarily in that CISC:

A. Has fewer instructions

B. Uses only register-to-register operations

C. Supports complex instructions that may take multiple cycles

D. Has no microcode

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: CISC commonly supports richer complex instructions, potentially with multi-cycle execution.

COA20 associates CISC with complex operations, richer addressing and variable lengths, and rejects universal one-cycle latency claims. C therefore gives the matching tendency and permits, rather than requires, multi-cycle execution. The remaining claims either reverse the usual complexity contrast or impose unsupported only/no restrictions. The answer does not claim modern RISC instructions can never take multiple cycles.

Why each choice is right or wrong

A - Incorrect: Fewer instructions reverses the usual simple-versus-rich instruction-set contrast rather than stating the note's CISC tendency.

B - Incorrect: Only register-to-register operations contradicts the richer addressing modes associated with CISC in the note.

C - Correct: Complex operations and possible multi-cycle latency fit the CISC tendency without claiming exclusivity of multi-cycle execution.

D - Incorrect: The note does not define CISC by absence of microcode; control implementation is a separate design dimension.

Primary note proof

Study-note section COA20

> Textbook RISC emphasizes simple operations, regular encodings, load/store memory access, plentiful registers and pipeline-friendly control.

Study-note section COA20

> CISC commonly supplies more complex operations, richer addressing modes and variable lengths.

Study-note section COA20

> Modern RISC instructions can take multiple cycles and modern CISC processors can translate instructions into simpler internal operations.

Question sources

Computer Organization & Architecture review questions · Question 36

Question 171 | NS0403

A RISC processor typically has more _____ compared to a CISC processor to compensate for simpler instructions:

A. Clock cycles per instruction

B. Addressing modes

C. Registers

D. Instruction bytes

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: Registers.

COA20 explicitly lists plentiful registers among textbook RISC characteristics, alongside load/store access and simple operations. C is therefore the supported tendency. More addressing modes and instruction bytes are not the stated compensating mechanism, and more cycles per instruction is not a defining guarantee. The typically qualifier matters because the labels alone do not impose a register count on every implementation.

Why each choice is right or wrong

A - Incorrect: More cycles per instruction is not the plentiful-register mechanism or a guaranteed RISC property.

B - Incorrect: Richer addressing modes are associated with CISC tendencies, not the stated RISC compensating feature.

C - Correct: Plentiful registers are explicitly one of the textbook RISC traits named by the complete note.

D - Incorrect: More instruction bytes is not the feature identified as supporting simple RISC operations.

Primary note proof

Study-note section COA20

> Textbook RISC emphasizes simple operations, regular encodings, load/store memory access, plentiful registers and pipeline-friendly control.

Study-note section COA-S04

> RISC means Reduced Instruction Set Computer; CISC means Complex Instruction Set Computer.

Question sources

Computer Organization & Architecture review questions · Question 37

Question 172 | NS0404

In Programmed I/O, how does the CPU know when the I/O device has finished its operation?

A. The I/O device sends an interrupt signal

B. The CPU periodically polls (checks) the status register

C. DMA controller notifies the CPU

D. The operating system schedules a check

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Poll the status register.

COA30 defines programmed/polled I/O as repeatedly checking device status while the CPU moves each item. B names that readiness/completion checking mechanism. Interrupt notification belongs to interrupt-driven I/O, and a DMA controller is a separate transfer mechanism. The OS may arrange when polling code runs, but that broad scheduling statement does not replace the actual status check.

Why each choice is right or wrong

A - Incorrect: An interrupt signal names interrupt-driven notification rather than programmed I/O's polling mechanism.

B - Correct: The CPU reads device status repeatedly to determine readiness or completion in programmed I/O.

C - Incorrect: DMA-controller completion notification belongs to a DMA transfer rather than programmed CPU-driven I/O.

D - Incorrect: Scheduling a check is broader software organization; polling the status register is the actual completion-detection method.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Question sources

Computer Organization & Architecture review questions · Question 38

Question 173 | NS0405

The main advantage of Interrupt-Driven I/O over Programmed I/O is:

A. Higher data transfer speeds

B. The CPU can perform other tasks while waiting for I/O

C. It requires no special hardware

D. It eliminates the need for device drivers

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: The CPU can do other work while the device operates.

COA30 says interrupt-driven I/O lets the CPU do other work until the device requests service. That removes continuous status polling, which is the principal advantage in B. Service still consumes CPU time and needs device support/software. Therefore it does not universally increase device transfer speed, eliminate hardware requirements or remove drivers.

Why each choice is right or wrong

A - Incorrect: Avoiding polling does not by itself guarantee that the device or bus transfers data at a higher rate.

B - Correct: The CPU can execute other useful work between setup and the device's service request.

C - Incorrect: Interrupt notification requires suitable device/controller support; the note does not promise no special hardware.

D - Incorrect: A handler/driver must still service the device; interrupt-driven operation does not remove that software role.

Primary note proof

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA29

> A controller is hardware; a driver is software that operates it.

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Question sources

Computer Organization & Architecture review questions · Question 39

Question 174 | NS0406

Direct Memory Access (DMA) primarily improves performance by:

A. Increasing CPU clock speed

B. Allowing a controller to transfer data between an I/O device and memory without the CPU copying each word

C. Compressing data before transfer

D. Using a faster bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Allowing a controller to transfer data between an I/O device and memory without the CPU copying each word

DMA transfers a block directly between an I/O device and memory after CPU setup. Its performance benefit is reduced per-word CPU copying, leaving the processor available for other work while the controller performs the transfer. CPU configuration and completion handling remain necessary, and DMA still consumes memory bandwidth. It does not inherently raise the clock, compress data or require a faster bus.

Why each choice is right or wrong

A - Incorrect: DMA changes who performs transfer work, not the processor's clock frequency.

B - Correct: The controller moves the block after setup, avoiding a CPU copy operation for every transferred word.

C - Incorrect: Data compression is not part of the DMA definition or its stated per-word CPU-work reduction.

D - Incorrect: DMA may use the existing bus and does not inherently replace it with a faster one.

Primary note proof

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> DMA does not eliminate CPU involvement in configuration, ownership, address mapping or completion.

Study-note section COA30

> DMA reduces per-word CPU work but competes for memory bandwidth.

Question sources

Computer Organization & Architecture review questions · Question 40

Question 175 | NS0407

In Memory-Mapped I/O, how are I/O devices accessed?

A. Using special IN/OUT instructions

B. Through dedicated I/O ports separate from memory

C. By assigning memory addresses to I/O registers, accessed like regular memory

D. Via DMA exclusively

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Device registers occupy memory addresses and use suitable memory access instructions.

COA31 says memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable loads/stores. C matches, but like regular memory refers to address/instruction access rather than identical side effects: MMIO widths, ordering and status-read behavior still matter. Special IN/OUT and a separate port space instead describe isolated I/O; DMA is independent of address-space choice.

Why each choice is right or wrong

A - Incorrect: Special IN/OUT instructions describe isolated/port-mapped I/O rather than the defining memory-mapped access mechanism.

B - Incorrect: A separate port space is the isolated I/O arrangement, not device registers in memory address space.

C - Correct: MMIO places device registers at memory-space addresses accessed through suitable loads/stores, with device-specific side-effect rules.

D - Incorrect: MMIO specifies addressing and access, not an exclusive requirement to transfer all device data through DMA.

Primary note proof

Study-note section COA31

> Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Study-note section COA31

> Isolated/port-mapped I/O has a separate port space and special instructions such as IN/OUT.

Study-note section COA31

> MMIO must respect device access widths, ordering and side effects; a status read is not necessarily a harmless RAM read.

Question sources

Computer Organization & Architecture review questions · Question 41

Question 176 | NS0408

An I/O 'Write Command' instructs the I/O module to:

A. Read data from the peripheral and store it in a buffer

B. Check the device status for errors

C. Take data from the data bus and transmit it to the peripheral

D. Reset the peripheral device

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Transmit system data to the peripheral.

COA29 defines write/output commands as sending system data to the device. C matches that direction: data from the system bus is handed to the peripheral. Reading brings device data toward the system, status testing inspects readiness/errors, and reset is a control action. These are distinct command functions even when one controller implements them all.

Why each choice is right or wrong

A - Incorrect: Moving peripheral data into a system buffer is the input/read direction rather than write/output.

B - Incorrect: Checking readiness or errors is a test/status command rather than transferring output data.

C - Correct: A write/output command sends data from the system side to the peripheral device.

D - Incorrect: Reset is a device-control action, not the ordinary data-output direction defining a write command.

Primary note proof

Study-note section COA29

> Control commands start actions, test/status commands inspect readiness or errors, read commands bring device data toward the system, and write/output commands send system data to the device.

Question sources

Computer Organization & Architecture review questions · Question 42

Question 177 | NS0409

Which I/O technique is most efficient for transferring large blocks of data, such as disk reads?

A. Programmed I/O

B. Interrupt-Driven I/O

C. Direct Memory Access (DMA)

D. Memory-Mapped I/O

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: DMA for substantial block transfers.

COA30 explicitly recommends DMA for substantial blocks because it reduces per-word CPU work to setup and completion duties. This matches disk-like large transfers. Polling and interrupt transfer still involve CPU servicing of individual items, while memory-mapped I/O is an address-space design choice that may coexist with DMA. The answer assumes the substantial-block context, not a universal fastest mechanism for every tiny transfer.

Why each choice is right or wrong

A - Incorrect: Programmed I/O makes the CPU check status and move each item, adding substantial work for large blocks.

B - Incorrect: Interrupt-driven transfer avoids continuous polling but still consumes CPU service work for transferred items.

C - Correct: DMA replaces much per-item CPU work with block setup/completion, matching the substantial-transfer recommendation.

D - Incorrect: Memory mapping specifies device addressing and can coexist with DMA; it does not itself offload block movement.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> Choose it for substantial blocks; mapping devices to addresses is a separate design dimension.

Question sources

Computer Organization & Architecture review questions · Question 43

Question 178 | NS0410

SMP (Symmetric Multiprocessing) systems are characterized by:

A. Multiple different processors with specialized roles

B. Compatible peer processors sharing memory and I/O under one operating system

C. Processors with separate private address spaces communicating by messages

D. Independent networked computers without shared memory

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: Compatible peer processors sharing memory and I/O under one operating system

SMP gives processors peer roles with shared memory and I/O under an operating system. Compatibility and symmetric roles matter; every physical processor property need not be identical. Specialized asymmetric roles do not state that model, while separate address spaces and independent networked computers describe distributed organization. The alternatives compare coherent organizations without an impossible single-processor multi-chip description.

Why each choice is right or wrong

A - Incorrect: Specialized unequal processor roles describe an asymmetric organization rather than the peer-role model.

B - Correct: Peer roles with shared memory and I/O under one OS match the SMP definition.

C - Incorrect: Separate private address spaces and message communication describe distributed-memory organization.

D - Incorrect: Independent computers without shared memory do not have the stated SMP shared-memory organization.

Primary note proof

Study-note section COA37

> SMP processors share memory/I/O and have peer roles under an OS.

Study-note section COA37

> Distributed-memory systems communicate between separate address spaces.

Question sources

Computer Organization & Architecture review questions · Question 44

Question 179 | NS0411

Pipelining improves CPU performance by:

A. Increasing the clock speed of each stage

B. Executing one instruction at a time faster

C. Overlapping the execution stages of multiple instructions simultaneously

D. Using larger registers

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Overlap stages of different instructions.

COA35 defines pipelining by overlapping stages from different instructions and says it primarily improves throughput. C directly states that mechanism. It does not require faster stages, larger registers or a single instruction completing all work faster. Pipeline fill/drain, stalls and the slowest stage constrain actual gains, so overlap rather than guaranteed latency reduction is the supported explanation.

Why each choice is right or wrong

A - Incorrect: Pipeline performance does not require increasing each stage's clock speed; clock choice is limited by the slowest stage.

B - Incorrect: An instruction still spans several stages, so lower single-instruction latency is not the defining improvement.

C - Correct: Overlapping stages lets different instructions progress concurrently, increasing completed-work throughput.

D - Incorrect: Larger register width does not define stage overlap or directly explain pipelining's throughput benefit.

Primary note proof

Study-note section COA35

> A pipeline overlaps stages of different instructions; it primarily improves throughput.

Study-note section COA35

> The ideal steady-state completion rate is one instruction per cycle, while an individual instruction still spans multiple stages.

Question sources

Computer Organization & Architecture review questions · Question 45

Question 180 | NS0412

In an arithmetic pipeline for floating-point addition, which is the FIRST sub-operation performed?

A. Normalize the result

B. Add/subtract the mantissas

C. Align the mantissas

D. Compare the exponents

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer D: Compare exponents first.

COA10 orders floating addition as compare exponents, shift the smaller significand to align, add/subtract, normalize and round. Comparing exponents must precede alignment because their difference determines the required shift. Adding before alignment combines unequal place weights, and normalization happens after arithmetic. Therefore D is the first named sub-operation.

Why each choice is right or wrong

A - Incorrect: Normalization follows aligned addition/subtraction rather than initiating the floating-point addition sequence.

B - Incorrect: Significands must first be aligned to a common exponent before their values can be added correctly.

C - Incorrect: Alignment follows exponent comparison, which determines which significand to shift and by how much.

D - Correct: Exponent comparison is explicitly first and supplies the information needed for subsequent alignment.

Primary note proof

Study-note section COA10

> For floating addition, compare exponents, shift the smaller significand to align, add or subtract, normalize and round.

Question sources

Computer Organization & Architecture review questions · Question 46

Question 181 | NS0413

Which statement describes vector-instruction semantics rather than guaranteeing simultaneous execution of every array element?

A. Processes one element at a time faster

B. One vector instruction applies an operation across multiple data elements, possibly in lane-sized batches

C. Uses fewer transistors

D. Operates at a higher clock frequency

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: One vector instruction applies an operation across multiple data elements, possibly in lane-sized batches

A vector instruction describes an operation over a sequence of elements. Its semantics cover multiple data elements, but physical lanes can process those elements in batches instead of executing the entire array at one instant. The choice distinguishes vector from scalar processing without requiring simultaneous execution of every element. Faster individual elements, fewer transistors or a higher clock are implementation properties rather than definitions of vector processing.

Why each choice is right or wrong

A - Incorrect: Faster scalar element processing does not define vector semantics.

B - Correct: This states the multi-element instruction semantics while permitting physical batching.

C - Incorrect: A transistor-count reduction is not part of the definition.

D - Incorrect: A higher clock frequency is not required by vector semantics.

Primary note proof

Study-note section COA37

> A vector instruction describes an operation over a sequence of elements; physical lanes may process them in batches, not all in one instant.

Question sources

Computer Organization & Architecture review questions · Question 47

Question 182 | NS0414

For an ideal single-issue RISC pipeline in steady state after filling, with balanced stages and no stalls, what is the completion throughput?

A. One instruction per two clock cycles

B. One instruction per clock cycle

C. Two instructions per clock cycle

D. Depends on instruction complexity

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: One instruction per clock cycle

An ideal single-issue pipeline overlaps stages of successive instructions. After filling, it completes one instruction each cycle when stages are balanced and no hazards or other stalls interfere. An individual instruction still crosses several stages, so this is throughput rather than one-cycle instruction latency. Specifying single issue excludes a superscalar interpretation that could complete more than one instruction per cycle.

Why each choice is right or wrong

A - Incorrect: One completion per two cycles is below the stated ideal no-stall rate.

B - Correct: The ideal steady-state single-issue completion rate is one per cycle.

C - Incorrect: Two per cycle requires multiple-issue resources beyond the stated design.

D - Incorrect: The ideal rate follows from the specified single-issue pipeline assumptions.

Primary note proof

Study-note section COA35

> In an ideal k-stage single-issue pipeline, n instructions take k+n−1 cycles when stages are balanced and there are no stalls.

Study-note section COA35

> The ideal steady-state completion rate is one instruction per cycle, while an individual instruction still spans multiple stages.

Question sources

Computer Organization & Architecture review questions · Question 48

Question 183 | NS0415

Flynn's taxonomy classifies parallel computer architectures based on:

A. Number of processors and memory size

B. Instruction streams and data streams

C. Clock speed and bus width

D. Cache size and register count

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: Instruction streams and data streams.

COA37 explicitly defines Flynn classification by instruction and data streams. SISD has one of each, SIMD one instruction stream with multiple data elements, and MIMD multiple streams of each. Processor count, capacity, clock and register/cache sizes are implementation properties rather than the two classification axes. Therefore B directly matches the taxonomy.

Why each choice is right or wrong

A - Incorrect: Processor count and memory size do not define the instruction-stream/data-stream axes of Flynn’s taxonomy.

B - Correct: Flynn classifies the number of instruction streams and data streams, producing SISD/SIMD/MISD/MIMD categories.

C - Incorrect: Clock and bus width measure implementation characteristics rather than Flynn’s stream-count classification.

D - Incorrect: Cache capacity and register count are not the two axes used by Flynn’s taxonomy.

Primary note proof

Study-note section COA37

> Flynn classifies instruction and data streams: SISD has one of each; SIMD applies one instruction stream to multiple data elements; MIMD uses multiple instruction and data streams; MISD has multiple instruction streams for a common data stream and is uncommon.

Question sources

Computer Organization & Architecture review questions · Question 49

Question 184 | NS0416

The primary goal of parallel processing is to increase:

A. Memory capacity

B. Clock frequency

C. Throughput - the amount of processing per unit time

D. Register size

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Throughput: completed processing per unit time.

COA37 says more cores increase available parallel work while serial dependencies and coordination limit speedup. COA34 defines throughput as completed work per unit time. Parallel processing aims to raise that work rate, matching C, rather than inherently increasing physical memory, clock frequency or register width. Actual gains still depend on enough independent work and manageable overhead.

Why each choice is right or wrong

A - Incorrect: Parallel execution does not inherently increase the system's installed memory capacity.

B - Incorrect: Additional concurrent work can improve throughput without increasing any processor's clock frequency.

C - Correct: Concurrent useful work aims to increase completed processing per unit time, subject to serial and coordination limits.

D - Incorrect: Register width is an architectural data-size property, not the primary objective of parallel execution.

Primary note proof

Study-note section COA37

> More cores increase available parallel work, but serial dependencies and coordination limit speedup.

Study-note section COA34

> Throughput counts completed work per unit time; latency measures one job's delay.

Question sources

Computer Organization & Architecture review questions · Question 50